Diogo Filipe Alves Ferreira

Licenciatura em Ciências da Engenharia Mecânica

Automatização do projeto de ferramentas progressivas para minimização dos tempos de desenvolvimento

Dissertação para obtenção do Grau de Mestre em Engenharia Mecânica

Orientador: Jorge Joaquim Pamies Teixeira, Prof. Catedrático, FCT-UNL
Co-orientador: Armando Bastos, Engenheiro responsável na MCG

Júri:

Presidente: Prof. Doutora Carla Maria Moreira Machado, Professora Auxiliar da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa.

Vogal: Prof. Doutora Teresa Leonor Ribeiro Cardoso Martins Morgado, Professora Auxiliar da Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2016

[Automatização do projeto de ferramenta progressiva]

Aos meus pais,
avós, irmão,
namorada
e amigos,

Agradecimentos

Gostaria de exprimir os meus mais sinceros agradecimentos a todos aqueles que direta ou indiretamente contribuíram para que esta etapa fosse concluída.

Em primeiro, como não poderia deixar de ser, um "muito obrigado" aos meus pais, que sempre me proporcionaram condições para que eu pudesse alcançar mais uma meta e que sempre se sacrificaram em prol do meu bem-estar, ao meu irmão que sempre me conseguiu distrair quando o *stress* apertava e à minha namorada, que sempre me ajudou e me apoiou nos momentos mais turbulentos.

Agradeço também a todos os meus colegas que me aturaram durante estes 5 anos, onde partilhamos experiências nunca mais esquecidas, bem como a todos os professores que me transmitiram algum do seu conhecimento.

Aproveito também para agradecer ao Engenheiro Armando bem como a todos os colaboradores da MCG que possibilitaram o desenvolvimento desta dissertação em ambiente empresarial.

Por fim não posso deixar de agradecer ao meu orientador, o professor Jorge Pamies Teixeira, por toda a paciência requerida ao longo desta dissertação e pelo facto de nunca me ter deixado "desistir" quando as dificuldades pareciam inultrapassáveis,

A todos vós um grandíssimo OBRIGADO!

Resumo

A crescente necessidade das empresas em aumentar a produtividade, reduzir os tempos de resposta assim como os custos associados a cada projeto, tem colocado os gabinetes de projeto numa pressão constante.

O objetivo desta dissertação foi a criação de uma metodologia de utilização das ferramentas informáticas existentes na empresa MCG com vista à automatização da atividade de projeto destas ferramentas progressivas, automatizando o desenvolvimento e a modelação dos elementos de características funcionais idênticas. Caso a automatização introduza uma redução significativa e economicamente viável, é proposto a automatização de todos os projetos.

Neste sentido e para contribuir para a minimização do tempo de projeto de uma ferramenta progressiva, e em parceria com a empresa *MCG-automotivo,* recorreu-se ao programa adotado na mesma, *CATIA versão5* e criou-se uma metodologia com o intuito de reduzir os tempos de projeto para que o impacto económico seja substancialmente inferior, sem que este prejudique as características mecânicas finais do produto.

Com esta dissertação foi possível reduzir o tempo de projeto da ferramenta progressiva em 20% comprovando assim a aplicabilidade da automatização em todos os projetos existentes na MCG.

Palavras-chave: Ferramenta progressiva, produtividade, CATIA, metodologia, automatização, programação.

Abstract

The increased need for companies to uplift their productivity, reducing response time and also the associated costs for each project; have been putting the responsible chains of command in a constant pressure.

The goal of this dissertation was created on a methodology based on the use of existing technological tools that were available in the worksite, with the purpose of automating every single aspect, from the development, to the creation of the elements that are characteristic identical and functional. If the automation introduces a significant reduction on response time and be economically reliable, it's imperative the automation in every single project.

On the effort, to try to contribute in the decrease of response time for the creation of a progressive tool project, and with partnership from MCG-Automotive resorting to their adopted software, CATIA V5, it was able to create a methodology with the purpose of reducing project time, that sub-consequently could decrease the economic impact without harming the mechanical characteristics in the final product.

With this dissertation it was possible to reduce the time on the creation of the progressive tool project by 20%, proving the applicability of automating all the existing projects on MCG.

Keywords: Progressive Tools, Productivity, CATIA, Methodology, Automatization; Programming.

Índice de Matérias

Índice de Imagens

Índice de Tabelas

Lista de Siglas

3D	Three Dimensional
CAD	Computer Assisted Design
CAE	Computer Assisted Engineering
CAM	Computer Assisted Manufacturing
CATIA	Computer Assisted Three Dimensional Interactive Application
DAC	Desenho Assistido por Computador
DS	Dassault Systèmes
MCAD	Multi-CAD Management
MCG	Manuel da Conceição Graça, Lda.
PDD	Progressive Die Design
SDRC	Structural Dynamics Reserch Corporation
RADE	Rapid Application Development Environment

1

Introdução

O Desenho Assistido por Computador (*DAC*, ou no inglês *CAD*) é muito importante nos dias que correm sendo que é o nome dado a sistemas computacionais (*software*) utilizados na engenharia para facilitar o projeto e desenho técnico de peças/componentes que posteriormente vão ser maquinados e/ou encomendados para um certo fim.

Estes sistemas fornecem uma série de ferramentas para construção de entidades geométricas planas (linhas, curvas, polígonos,…) ou mesmo objetos tridimensionais (cubos, esferas,…). Além disso também disponibilizam ferramentas para relacionar essas entidades ou objetos, como por exemplo raios de concordância (*filets*) entre 2 faces não-coincidentes, ou subtrair formas de dois objetos tridimensionais para obtenção de um terceiro.

Contudo, hoje em dia as empresas procuram mais a capacidade que estes *softwares* têm de ser automatizados, minimizando o tempo de desenvolvimento do produto, tornando-se mais competitivas em relação à concorrência. Em praticamente todas as aplicações disponíveis no mercado, existem funcionalidades que permitem através da programação, gerar um conjunto mais ou menos alargado de ações de modelação do produto.

É, devido a esse fator, importante realizar um estudo destes programas e determinar aquele que melhor se adequa aos objetivos desta dissertação, tendo como foco principal a modelação e programação de componentes a três dimensões.

Esta dissertação foi desenvolvida em parceria com a empresa MCG, que se dedica à produção de componentes metálicos para vários clientes. Deste modo foi possível por em prática conhecimentos previamente adquiridos ao longo do curso de Engenharia Mecânica.

Devido à conjuntura económica que atravessamos nos dias de hoje é de vital importância a redução de custos, a redução de prazos de entrega ao cliente e ainda o aumento da produtividade, isto mantendo sempre a qualidade do produto final evitando assim denegrir a imagem que a empresa MCG

conseguiu estabelecer ao longo do tempo. Este trabalho é uma excelente oportunidade para mostrar que se consegue reduzir estes fatores sem nunca por em causa a fiabilidade do produto final, bastando para isso alterar apenas algumas das metodologias de trabalho até aqui usadas.

Esta dissertação pretende focar-se na resposta a vários requisitos importantes para a empresa:

- Minimização dos tempos de projeto de ferramentas progressivas
- Redução dos custos totais de projeto
- Aumento da qualidade do serviço

Estes objetivos pretendem ser atingidos recorrendo à parametrização/automatização do projeto dos componentes de um projeto, usando os recursos existentes, no caso, um programa *CAD*, através da programação do mesmo, desenvolvendo-se uma nova metodologia de projeto.

Como o negócio da empresa MCG não se prende com um tipo *standard* de peças, optar-se-á por realizar a conceção do projeto de uma ferramenta progressiva de uma peça metálica com características mais amplas possíveis, para que deste modo abranja a maior diversidade de operações necessárias.

Desde 2010, Portugal está em crise económica que abrange todos os sectores de atividade, muito particularmente o sector industrial.

Sendo este o cenário que a empresa MCG enfrenta é de vital importância o desenvolvimento de novas metodologias de trabalho, investindo no conhecimento dos seus colaboradores bem como ter a capacidade de adaptação às exigências de mercado. Para a competitividade da empresa também é necessário manter a qualidade dos componentes finais, reduzir o tempo e projeto e produção para que se possa reduzir o seu custo final.

Será tudo isto possível recorrendo à capacidade que os *softwares* atuais disponibilizam? Faremos primeiro um pequeno levantamento dos principais programas existentes no mercado.

2

Contextualização da Empresa

A *MCG automotive* é a principal unidade de negócio da empresa *MCG*, localizada no Carregado, Portugal, que conta já com mais de 60 anos de história, dedicada à indústria de componentes metálicos. Dedicando-se ao *design* e ao fabrico de ferramentas de estampagem, tem como principais clientes grandes empresas tais como *Ford*, *Opel*, *Bosch* ou ainda *Volkswagen*.

A *MCG mind for metal* dispõe de 5 unidades industriais todas localizadas no seu polo no Carregado. Estas unidades dão corpo à estratégia de diversificação que desde de 2010, a empresa tem seguido, reunindo um conjunto de tecnologias e competências que permitem dar resposta aos mercados exigentes onde está presente. As cinco unidades existentes são divididas e enumeradas de acordo com a sua função específica na área de negócio da MCG: Metal 1 (Estampagem, Soldadura, Montagem, Tratamento de Superfície), Metal 2 (Design e fabrico de ferramentas de estampagem), Metal 3 (Prototipagem e fabrico rápido), Metal 4 (Estampagem) e Metal 5 (Área de montagem).

Esta dissertação incide sobretudo na Metal 2, local onde é feito o projeto de uma ferramenta progressiva.

A metodologia de trabalho de uma empresa influencia em grande parte o rendimento desta, obrigando a uma utilização inteligente dos recursos necessários disponíveis, para não perder competitividade a médio/longo prazo. Atualmente, o processo de criação de um novo produto na MCG pode sumariar-se pelo fluxograma patente na imagem 2-1.

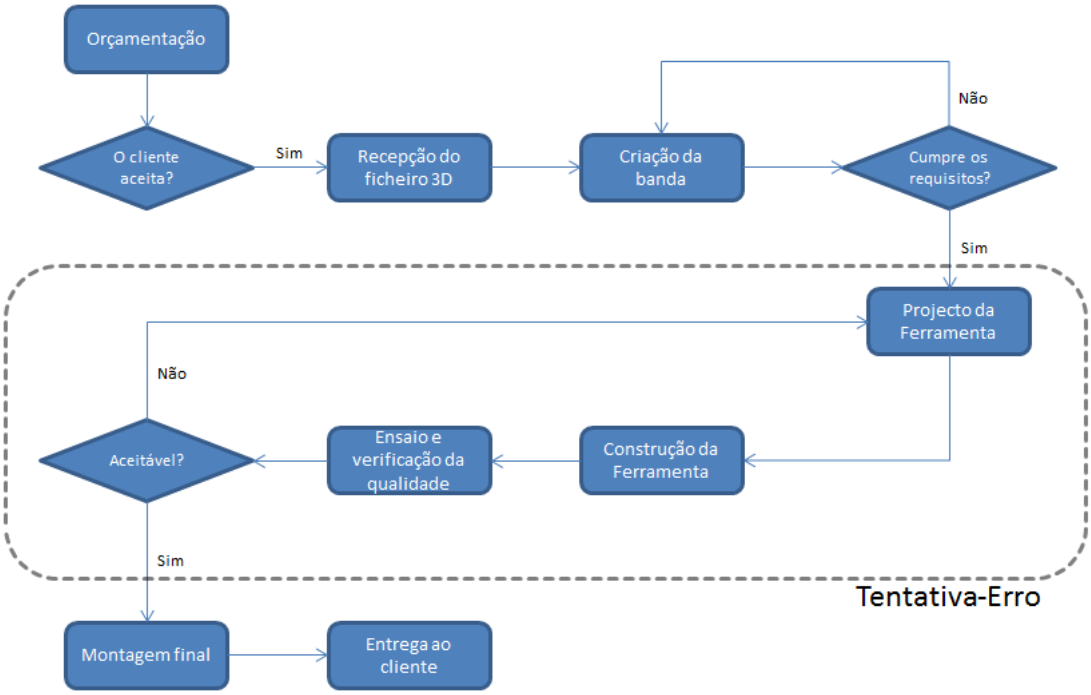


Imagem 2-1 Fluxograma da metodologia de trabalho utilizada na MCG

A metodologia inicia-se pela orçamentação, etapa onde é levantado o tempo de projeto e os custos associados. A aceitação do orçamento pelo cliente dá início ao projeto. Envia-se ao gabinete de projeto o ficheiro *CAD 3D* do componente a desenvolver, bem como todas as suas especificações, entre as quais a capacidade da prensa em que a ferramenta irá trabalhar, fator importante na sua automatização.

Com o estudo da peça procede-se à planificação do artigo a desenvolver numa banda até que os requisitos funcionais estejam atingidos. Logo após a sua aceitação dá-se início ao projeto da ferramenta progressiva propiamente dita. Este é um processo que engloba também a construção da ferramenta e os respetivos ensaios de qualidade, num ciclo tipicamente "tentativa-erro", representados na figura 2-1 com traçado interrompido. Este processo, tal como o nome sugere, é iterativo e só estará concluído após obtenção dos requisitos dimensionais exigidos pelo cliente. É a diminuição do tempo inerente a este processo que esta dissertação visa reduzir, minimizando o tempo existente no projeto de uma ferramenta progressiva.

3

Estado da Arte

O projeto de uma ferramenta progressiva pela via tradicional é um processo que se pode tornar longo e demorado. Cada projeto pode requerer um processo criativo para aplicar novas ideias e representar um desafio para o projetista. Contudo, é também um processo enfadonho de cálculos detalhados e tentativas de esboço sem resultados satisfatórios.

Uma ferramenta progressiva é um sistema complexo de elementos combinados para que no seu conjunto produzam peças em chapa de variadas características. Trata-se de ferramentas que, não obstante, produzirem peças muito diversas, possuem um conjunto de elementos com características funcionais idênticas. A produção de peças metálicas estampadas através destes tipos de ferramentas carece dos seguintes componentes:

- A ferramenta progressiva propriamente dita
- Uma prensa onde a ferramenta irá trabalhar
- Um alimentador de chapa automático
- Um dispersor automático de óleo para haver a devida lubrificação

A correta sincronização destes componentes é fundamental para a obtenção da peça final. O funcionamento de uma ferramenta progressiva pode ser descrito pelos seguintes passos:

1) A banda avança na ferramenta uma distância equivalente ao valor do passo da ferramenta, por ação do alimentador automático
2) A prensa desce até ao seu ponto morto inferior, permitindo assim a execução das várias operações de corte, dobra e ou estampagem
3) A prensa volta a subir até ao ponto morto superior
4) O alimentador faz com que a banda avance mais um passo
5) O ciclo repete-se

A decomposição da ferramenta progressiva nos seus elementos mais básicos pode ser a esquematizada na imagem 3-1, tendo em conta que a construção é feita da banda para o exterior:

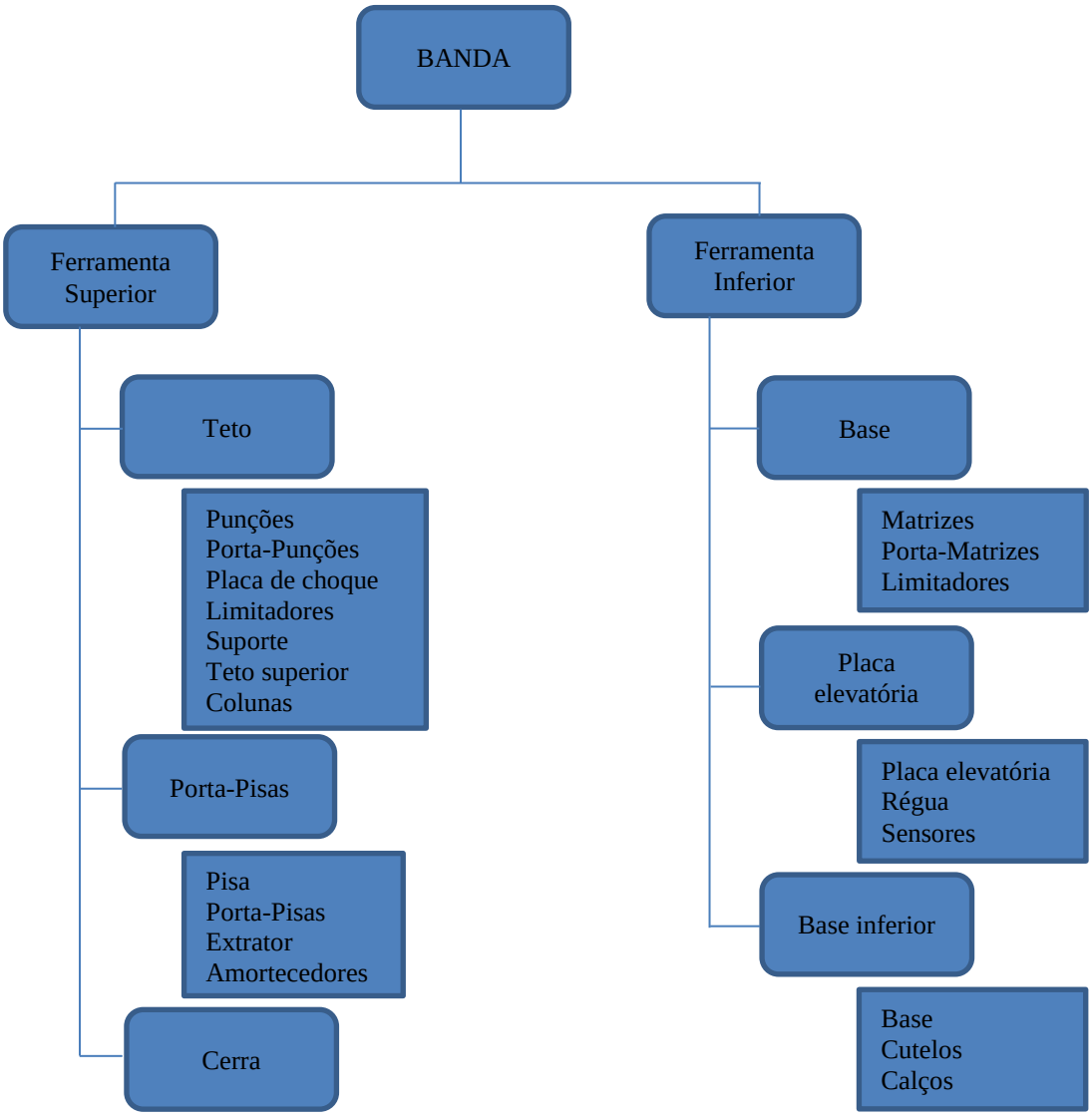


Imagem 3-1 Fluxograma da constituição de uma ferramenta progressiva

A peça normalmente não está projetada para o fabrico, mas sim para a função que irá executar, sendo por isso que o ponto de partida para o projeto de uma ferramenta progressiva tende a passar por um redimensionamento da peça a fabricar, estando este ao critério do projetista.

Uma grande mais-valia é a existência de *softwares* que fazem a planificação da peça e a pré-construção de uma possível ferramenta progressiva, contudo, e tendo em conta que cada empresa segue as suas próprias normas, estes programas poderão não representar um valor acrescentado no desenvolvimento e automatização deste tipo de projeto.

É nesta vertente que surgem estudos para automatizar o mais que possível o projeto indo ao encontro das necessidades de cada empresa, em que os *softwares* em uso são muito diversificados, designadamente *NX*, *Creo*, *CATIA*, entre muitos outros.

Embora este tema já tenha sido debatido em outras dissertações (como as apresentadas na bibliografia), o novo debate do mesmo, desta vez para o caso próprio da Manuel da Conceição Graça é sempre relevante, pois cada programador tem a sua forma de pensar e criar algoritmos, sem que estas sejam exatamente iguais às que foram desenvolvidas em dissertações anteriores.

Na criação de um projeto o projetista pode optar por duas variantes: elaboração de um modelo paramétrico e/ou optar por uma modelação direta.

Na primeira opção recorre-se ao uso de parâmetros para caracterizar os diferentes constituintes que fazem parte do projeto, destacando-se entre eles textos, relações, dependências, equações matemáticas ou ainda ciclos de programação. Na sua utilização pretende-se atingir a precisão do projeto e a sua total definição, sendo para isso necessária a elaboração de uma estrutura de projeto que se adapte a estes parâmetros. No sentido de se conseguir aplicar este tipo de modelo é necessário a aquisição de elevados conhecimentos teóricos e experiência por parte do projetista. Deste modo e com recurso a vários parâmetros tem-se como objetivo que o utilizador tenha a mínima intervenção possível dentro do projeto, onde a única necessidade está em alterar previamente as variáveis. Esta experiência por parte do projetista deve ser injuntiva na previsão dos parâmetros essenciais e das suas variações numa possível reutilização do projeto.

Relativamente à modelação direta esta prende-se com o objetivo de conceber de forma livre e rápida uma geometria e desta forma dar resposta às futuras implicações que possam advir nas decisões de um projeto de engenharia. Neste tipo de modelação, a facilidade na reutilização de um projeto previamente desenvolvido, torna-se bastante complexo, visto a falta de dependências ou relações nos componentes criados. Em contrapartida a utilização deste tipo de modelação é mais percecionada e flexível, muitas vezes associada a utilizadores pouco experientes de forma a criarem bons suportes de modelação.

3.1 Caracterização de *softwares*

Neste subcapítulo tem-se como objetivo descrever os principais *softwares* selecionados de modo a que se consiga determinar a melhor opção para a aplicabilidade da automatização do projeto de ferramentas progressivas. Selecionaram-se três *softwares* normalmente utilizados na empresa distintos: o *CATIA* da *Dassault Systèmes*, o *NX* da *Siemens* e o *Solidworks* também da *Dassault Systèmes*.

O objetivo é então caracterizar individualmente estes programas na sua generalidade e na vertente da programação para uma futura comparação no sentido de se determinar se o *software* existente possui as funcionalidades adequadas para os objetivos propostos.

3.1.1 *CATIA*

Líder de mercado atualmente o *Computer Aided Three Dimensional Interactive Application* (*CATIA*) foi desenvolvido pela *Dassault Systèmes* desde 1981 encontrando-se atualmente na versão V6. É um sistema de multiplataforma para *CAD*, *CAE* e *CAM*. O *CATIA* possui diversas vertentes onde é possível a criação e/ou modificação de projetos, destacando-se entre outras a modelação de superfícies, o processamento de chapas, projetos *CAM* e ainda projetos de engenharia como modelação de volumes.

A sua programação é possível graças aos recursos: *macros*, *knowledgeware*, *design tables* e *Rapid Application Development Environment* (*RADE*). Uma *macro* consiste na gravação dos passos e comandos realizados, que posteriormente poderão ser invocados por um único comando. Em alternativa, a *macro* pode ser criada utilizando a linguagem de programação (*script*) do *CATIA*. Esta pode ser feita em dois formatos diferentes: recorrendo ao *CATScript*, onde se cria uma linha de código numa linguagem básica (*basic script*) que é posteriormente gravada em formato "*catscript*"; ou em *Visual Basic Script*, onde é utilizada a linguagem comum de *Visual Basic* e que contem um corretor de erros (*debugger*). Esta última apenas é executável no sistema operativo *Windows*. A existência de um corretor de erros de programação é uma enorme vantagem em relação aos outros métodos, uma vez que permite a rápida deteção de erros na elaboração das linhas de código. Na imagem 3-1 exibem-se exemplos das duas linguagens de programação existentes no *CATIA*.

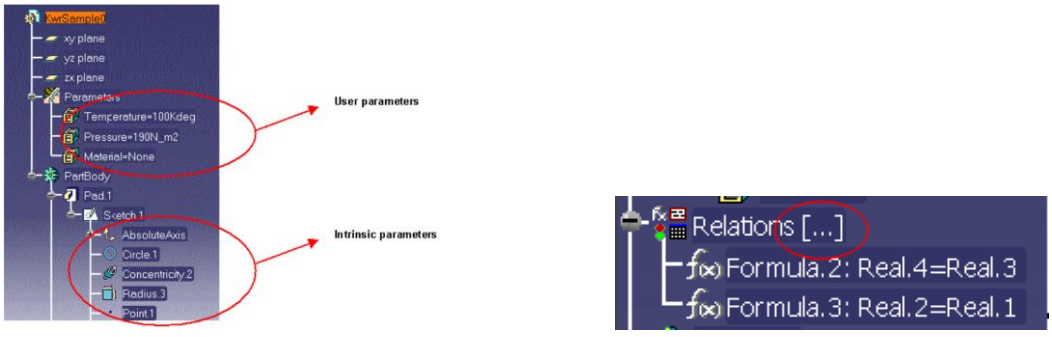
```
'Script by Nick
'Last revised November 4th, 2011
'This macro takes a screen shot with a white background and saves it in a folder
'=====================================================
Sub CatMAIN()

Dim ObjViewer3D As Viewer3D
Set objViewer3D = CATIA.ActiveWindow.ActiveViewer

Dim objCamera3D As Camera3D
Set objCamera3D = CATIA.ActiveDocument.Cameras.Item(1)

'Input box to name the screen capture image file
Dim partname As String
partName = Inputbox ("Please name the image.")

If partName="" Then

MsgBox "No name was entered. Operation aborted.", vbExclamation, "Cancel"

Else
```

```
Sub CATMain()

Dim partDocument1 As PartDocument
Set partDocument1 = CATIA.ActiveDocument

Dim part1 As Part
Set part1 = partDocument1.Part

Dim hybridBodies1 As HybridBodies
Set hybridBodies1 = part1.HybridBodies

Dim hybridBody1 As HybridBody
Set hybridBody1 = hybridBodies1.Add()

part1.Update

Dim hybridShapeFactory1 As HybridShapeFactory
Set hybridShapeFactory1 = part1.HybridShapeFactory

For I = 0 To 360 Step 10

Dim hybridShapePointCoord1 As hybridShapePointCoord
Set hybridShapePointCoord1 = hybridShapeFactory1.AddNewPointCoord(Sin(I) * 100, Cos(I) * 100, 0

hybridBody1.AppendHybridShape hybridShapePointCoord1

part1.InWorkObject = hybridShapePointCoord1

Next

part1.Update

End Sub
```

a) b)

Imagem 3-1 a) Ilustração do menu de uma *macro* realizada em *CATScript*; b) Ilustração de uma *macro* realizada em *VBAScript*.

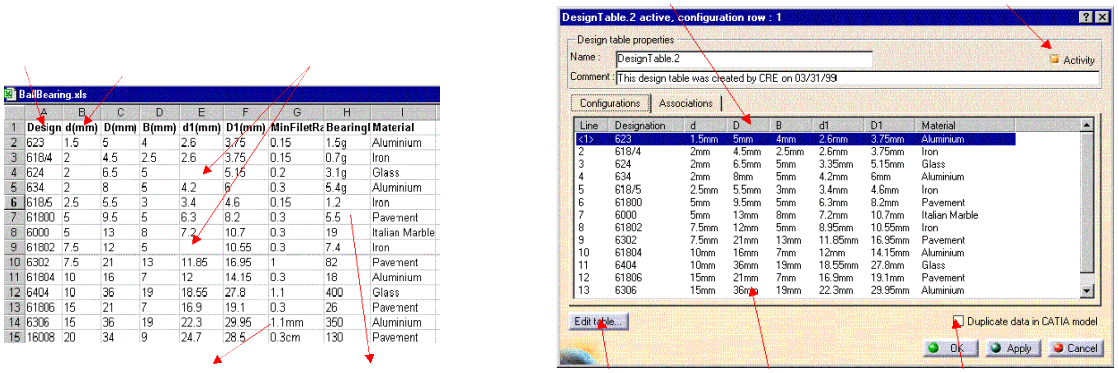
Relativamente ao *knowledgeware* trata-se de um nível de programação mais avançado que recorre a parâmetros solicitados ao projetista para condicionar e restringir a modelação, como por exemplo uma limitação ao nível do volume da peça que se está a projetar. Existem duas origens para estes parâmetros sendo uma delas definida por padrão (*intrinsic defined*) e outra definida pelo utilizador (*user defined*). Das imposições definidas por padrão, criadas pelo próprio *CATIA*, destacam-se as perpendicularidades entre linhas, concentricidades, ou ainda imposições de geometrias. Estes parâmetros podem ser todos importados de um ficheiro externo ao *CATIA*, como por exemplo o *Excel* da *Microsoft*. Estes constrangimentos acrescentam informação ao projeto que pode ser desenvolvida através de fórmulas ou tabelas. Um exemplo prático de como esta ferramenta consegue ser poderosa é quando queremos que uma dimensão esteja diretamente relacionada com uma outra já existente e onde o utilizador pode interferir: na figura 3-2 b) podemos observar que o valor de uma certa variável denominada “Real.4” possui o mesmo valor do que a variável “Real.3”, sendo apenas possível alterar esta última.



a) b)

Imagem 3-2 a) Ilustração dos parâmetros relativos ao *knowledgeware*; b) Exemplo de parametros introduzidos pelo utilizador

Adicionalmente e não menos importante, destacam-se as *design tables*, que permitem a criação de componentes pertencentes a uma mesma família, com diferentes configurações. Um exemplo de onde as *design tables* são úteis é na criação de várias combinações de um mesmo componente, onde são alteradas algumas dimensões, tal como mostra a figura 3-3. Neste caso está exemplificado uma lista de rolamentos com a mesma geometria sendo que os valores que variam são os seus diâmetros, a sua largura e o tipo de material que são produzidos. As *design tables* podem ser importadas de três formatos distintos: *Excel*, "*.txt*" e *CATIA Design Table*.



a) b)

Imagem 3-3 a) Ilustração de uma folha *Excel* para posterior importação para o *CATIA*; b) Ilustração de uma *Design Table* do *CATIA*

Finalmente, existe ainda a possibilidade de recorrer ao *RADE*, que é o ambiente de programação da *Dassault Systèmes* onde foi desenvolvido o *CATIA*, e talvez por isso, não pertence ao pacote base deste *software*. Nesta vertente o utilizador não fica limitado à linguagem de programação "*C++*", sendo possível programar também em *CAA V5*. Contudo a utilização deste sistema de programação apenas é desenvolvido por encomenda por parte dos clientes à *Dassault Systèmes*.

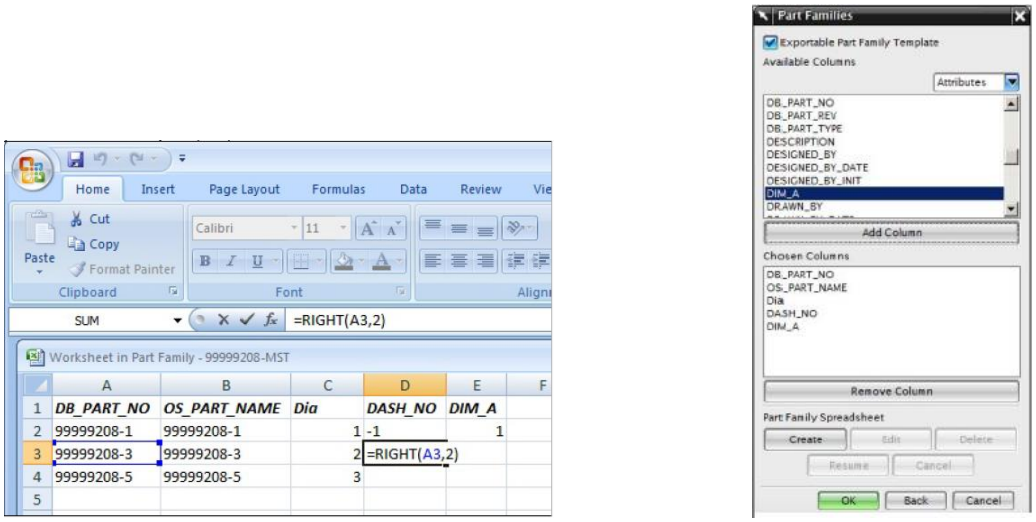
3.1.2 *NX*

Em 2002 a empresa *Siemens* apresentou uma solução melhorada do *software Unigraphics*, também com funcionalidade de *CAD*, *CAE* e *CAM* com a designação de *NX*. Incluiu posteriormente o *ST Synchronous*, que possibilita a criação de novos componentes através do projeto direto, rivalizando assim com os *softwares* existentes na altura. Possui, entre muitas outras funcionalidades, o *Multi-CAD Management* (*MCAD*) que lhe confere a capacidade de criar novos componentes, gerir modelações e/ou reutilizar projetos desenvolvidos por outros *softwares* como por exemplo o *CATIA*, *Solidworks* ou *AutoCAD*. O *NX* alega possuir uma tecnologia de 4ª geração com o intuito de diminuir o custo

computacional e o aumento da performance do projeto, onde existe a possibilidade de vários utilizadores efetuarem alterações das várias peças, que constituem um projeto, ao mesmo tempo, sendo estes notificados sempre que haja uma atualização.

A programação no *NX* recorre a uma de três formas possíveis: *macro*, *family tables* ou *journal's*. A criação de *macros* é em tudo análogo à já explicada acima, a respeito do *CATIA*, quer a nível de gravação de *macros*, quer a nível de criação de raiz das linhas de código. Contudo, neste *software* existe a carência de um *debugger*.

A metodologia para a criação de *family tables* é análoga à criação de *design tables* no *CATIA*, contudo neste caso, a atribuição de valores aos parâmetros escolhidos terá de ser feita numa folha de cálculo de *Excel* e só posteriormente é importada e criada uma árvore de produtos das alterações programadas. A imagem 3-4 representa uma folha de *Excel* e a sua importação no *NX*.



a) b)

Imagem 3-4 a) Ilustração da folha *Excel* com os parâmetros para posterior introdução no *NX*; b)Ilustração dos menos de escolha na *Famalie Tables*

Além desta funcionalidade, o *NX* possui módulos de auxílio ao projeto de moldes de injeção de plástico, de maquinagem e de ferramentas progressivas, mais usualmente denominado *Progressive Die Design* (*PDD*). No *PDD* ter-se-á de importar um modelo *3D* da peça a produzir, que pode estar em formato diferente daquele que o *NX* cria. Este reconhece o artigo como sendo uma chapa de metal, procedendo então à sua planificação. Neste processo, o *software* pode sugerir uma configuração de banda para minimizar o desperdício de chapa.

Depois de a banda ser aprovada pelo projetista, o programa simula a sequência de operações, permitindo, a partir dela projetar os elementos constituintes de uma ferramenta progressiva.

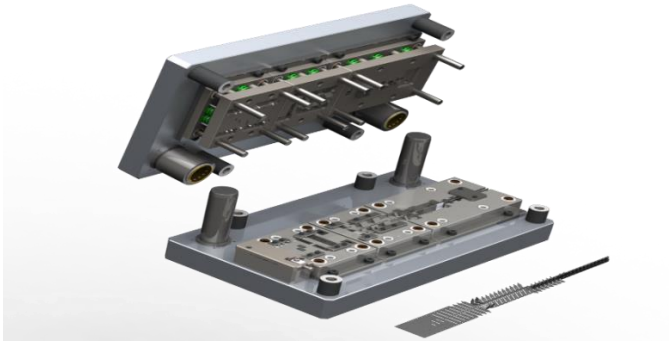


Imagem 3-5 Ilustração de uma ferramenta progressiva desenvolvida no *NX*

3.1.3 *Solidworks*

Foi em 1995, dois anos após ser fundada, que a empresa *Solidworks* lança para o mercado a sua primeira versão do programa de modelação 3D. A filosofia de base do *Solidworks* é distinta dos outros sistemas. A base é o sistema de modelação integrado num conjunto cada vez mais vasto de funcionalidades decorrentes de diferentes *softwares* criados pelos chamados *Solidwork-Partners*. É o caso do *SolidCam* para as funcionalidades de simulação de maquinação, ou ainda o *LogoPress3* que permite o projeto de uma ferramenta progressiva. Estas soluções são integráveis no *Solidworks* de tal forma que o utilizador toma a solução global como se se tratasse apenas de um *software*.

O *LogoPress3*, em particular, é um programa da *Accurate Die Design Inc*, permitindo, a partir da peça que se pretende desenvolver, projetar toda a ferramenta progressiva de uma forma eficaz e rápida. Na imagem 3-6 está retratada a criação de punções recorrendo ao *Logopress3*.

Imagem 3-6 Ilustração de punções criados através do *Logopress3*

Com a fita definida segue-se a criação da ferramenta progressiva propriamente dita, sempre do interior para o exterior, começando pelos punções. O projetista pode optar por realizar esta tarefa manualmente, ou definir a área afeta ao punção e o *software* cria o ficheiro *3D* correspondente. De seguida procede-se à definição/criação dos restantes elementos da ferramenta através da adição de uma estrutura genérica que tome em consideração não só a altura dos punções como também o comprimento da fita. É nesta estrutura que serão definidos os elementos fundamentais como as matrizes, bases e tetos. É sempre possível adicionar componentes em caso de necessidade como por exemplo placas de choque. Por último são adicionados os elementos de ligação como parafusos, cavilhas, batentes entre outros importados a partir de uma extensa biblioteca existente no programa. É de destacar que todos os módulos ou elementos que são criados têm a possibilidade de serem editados pelo utilizador.

4

Desenvolvimento

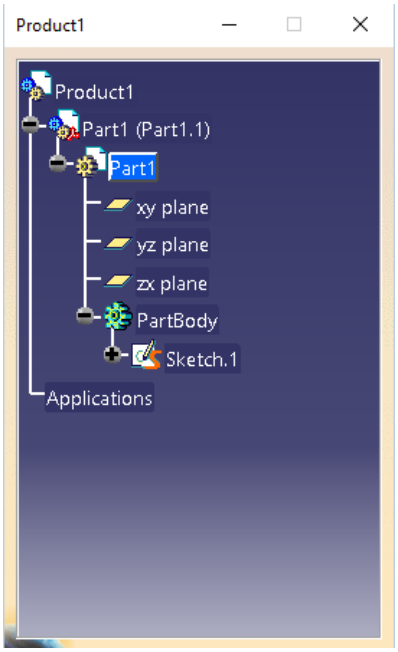
4.1 Seleção do *Software*

No capítulo 1 foi enquadrado o problema que se pretende solucionar: redução do tempo de projeto de uma ferramenta progressiva, aumentando assim a sua eficiência.

Entre as potencialidades descritas e o funcionamento da programação de diversos *softwares* expostos no capítulo 3, fica a questão de qual o melhor software para a obtenção dos objetivos propostos, sabendo que também é pretendido uniformizar o formato de projeto do gabinete para um único *software*, para assim evitar a compra de licenças desnecessárias.

Existe a necessidade de se optar por um *software* suficientemente versátil na capacidade de transmissão de informação entre as diversas plataformas de modelação mas que este seja aceite pelo cliente final sem que haja problemas de compatibilidade. Outro ponto importante é considerar as funções de programação disponíveis de cada *software*, quer ao nível da aquisição da licença, quer ao nível do custo associado à compra de módulos. Tal fator não serve como via de desempate visto que a *MCG* possui as licenças de ambos os programas. É importante destacar que tanto no *CATIA* como no *NX* há a necessidade de conhecimentos mais abrangentes de linguagem de programação para se conseguir retirar o maior proveito possível destes *softwares*.

Não havendo uma razão clara de utilização de um em detrimento do outro, optou-se por utilizar o *CATIA*, devido à existência do seu corretor de erros, *debugger*, que muito poderá ajudar na correta deteção de falhas nas linhas de código da programação. O *CATIA* trabalha muito recorrendo à sua árvore de produto, o que torna a sua programação mais intuitiva. Na imagem 4-1 é representada a seleção de um *sketch* na árvore do produto, bem como a sua linha de código respetiva.



a)

```
Sub CATMain()

Dim documents1 As Documents
Set documents1 = CATIA.Documents

Dim partDocument1 As PartDocument
Set partDocument1 = documents1.Item("Part1.CATPart")

Dim part1 As Part
Set part1 = partDocument1.Part

Dim bodies1 As Bodies
Set bodies1 = part1.Bodies

Dim body1 As Body
Set body1 = bodies1.Item("PartBody")

Dim sketches1 As Sketches
Set sketches1 = body1.Sketches

Dim originElements1 As OriginElements
Set originElements1 = part1.OriginElements

Dim reference1 As Reference
Set reference1 = originElements1.PlaneXY

Dim sketch1 As Sketch
Set sketch1 = sketches1.Add(reference1)

Dim arrayOfVariantOfDouble1(8)
arrayOfVariantOfDouble1(0) = 0#
arrayOfVariantOfDouble1(1) = 0#
arrayOfVariantOfDouble1(2) = 0#
arrayOfVariantOfDouble1(3) = 1#
arrayOfVariantOfDouble1(4) = 0#
arrayOfVariantOfDouble1(5) = 0#
arrayOfVariantOfDouble1(6) = 0#
arrayOfVariantOfDouble1(7) = 1#
arrayOfVariantOfDouble1(8) = 0#
Set sketch1Variant = sketch1
sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble1

part1.InWorkObject = sketch1

Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()
```

b)

Imagem 4-1 a) ilustração da árvore de produto; b) Ilustração do código para a criação do *sketch* representado em a).

Sempre que é necessário a introdução de um novo *sketch*, há a necessidade de o localizar ao nível do conjunto (denominado por *Product*), da peça (*Part*), do corpo da peça (*PartBody*) e do plano onde se quer trabalhar. Um exemplo simples é a criação de um *sketch* no plano XY de um determinado subconjunto. Primeiramente ter-se-á de ativar a edição da *Part*, denominada por *Part1*, procedendo-se de seguida à edição do *PartBody* para que, por fim, se possa adicionar o *sketch* pretendido no plano XY do referencial.

Com isto e conhecendo a linha de código para a realização de *extrudes* (denominados de *PAD* ou *POCKET*, consoante se esteja a adicionar material no sentido positivo ou retirar no sentido negativo, respetivamente) é possível elaborar qualquer componente. É de destacar que neste tipo de programação é necessário dar coordenadas a cada ponto, linha, ou circunferência criados, tal como é demonstrado no código completo, esquematizado no anexo A, da realização de uma matriz.

Sabendo o algoritmo que o programa executa e recorrendo aos ciclos de condições transversais a todas as linhas de programação, tais como o ciclo "*if*", "*for*" ou "*while*" é possível automatizar as macros para a obtenção dos objetivos previamente estabelecidos.

Se a hipótese apresentada demostrar uma efetiva redução do tempo de projeto face ao tempo investido na programação, sem qualquer diminuição da qualidade final do componente e do artigo a produzir, é justificada a implementação desta metodologia em todos os projetos a desenvolver.

4.2 Objeto de estudo

O objeto de estudo da presente Dissertação de Mestrado será uma peça metálica representada na imagem 4-2. A mesma é constituída por uma dobra a 90° com duas reentrâncias simétricas. Tendo uma espessura de 1,5 milímetros possui também dois furos sendo um deles uma simples circunferência de raio 3,1 milímetros (a) e outro semelhante a uma forma oval (b). Será produzida recorrendo a uma chapa metálica de largura 150 milímetros.

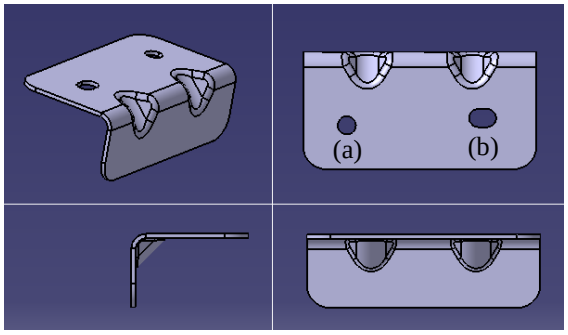


Imagem 4-2 Ilustração das diferentes vistas da peça a desenvolver

4.3 Descrição da programação

Neste subcapítulo descreve-se o processo de automatização do projeto da ferramenta progressiva para o projeto da peça em estudo através de vários códigos e imposições criados para atingir a programação desejada. Todavia, não é realizada uma descrição de todos os elementos programados, mas sim daqueles mais importantes ou os que apresentem uma programação diferente dos apresentados até então.

É possível criar um código de programação através de diferentes abordagens uma vez que o limite da programação é o conhecimento que o próprio programador possui. Para este trabalho foi escolhido realizar uma programação através do recurso ‘gravação de macros’ e posterior introdução de parâmetros que comandem os componentes vitais da ferramenta.

O projeto tem como elemento base um ficheiro de conjunto (*Product*), onde são adicionados todos os componentes desenvolvidos, gerando assim a ferramenta progressiva. Seguindo a divisão da

ferramenta sugerira na Tabela 3-1 inicia-se este projeto pela criação individual da banda e posterior desenvolvimento do resto da ferramenta progressiva.

O processo de automatização não deve ser aplicado às planificações uma vez que se assume que se tem como ponto de partida uma correta planificação da chapa. Naturalmente existem mais alternativas de programação sendo uma delas a criação de parâmetros para todas as dimensões do produto final. No entanto, esta metodologia possui fortes possibilidades de ser demasiado rígida na adaptação de novas configurações e pouco intuitiva na alteração de apenas um parâmetro. Daí não se optar por essa via. Se o projetista pretender alterar algo que não esteja definido através de *inputs*, poderá faze-lo diretamente no esboço (*sketch*) do componente. Como referido anteriormente, em anexo está disponível o código completo da criação de uma matriz para consulta.

4.3.1 Banda/Fita

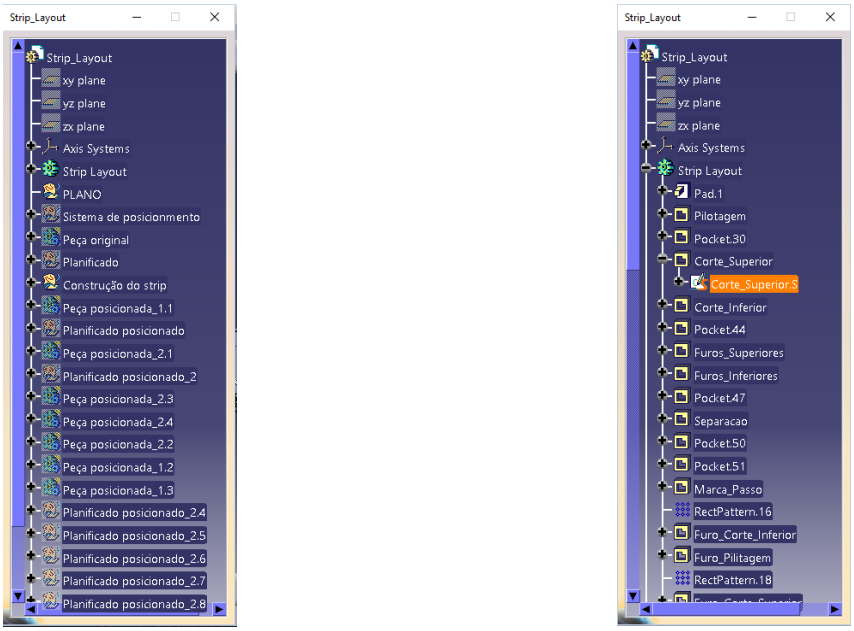
O primeiro componente a ser concebido num projeto de uma ferramenta progressiva é a banda, sendo que é a partir desta que a ferramenta ganha forma. Para a sua elaboração é essencial ter em atenção vários fatores, destacando-se entre eles a largura da banda, o seu passo e a sua espessura.

É no processo de orçamentação que a largura e o passo ficam definidos, para que o projetista possa, se possível, otimizar esses valores no decorrer da elaboração do projeto da banda/fita da ferramenta progressiva. Na MCG existem três *softwares* que podem ajudar a criação da banda: *Siemens NX*, *CATIA v5* e *Autoform*, existindo vários critérios para a elaboração da banda, mas que se dá sempre prioridade àquele que exige o menor número de passos possíveis, de modo a minimizar o tamanho da fita.

Após definida a banda, todos os outros componentes são projetados a partir das suas dimensões, daí considerar-se a fita o componente mais importante no que toca ao projeto de uma ferramenta progressiva. Seja programação de uma ferramenta progressiva ou de um sistema operativo de computadores, é necessário que exista uma metodologia a seguir, para que assim se minimizem os erros que possam surgir.

Para que a programação das macros seja o mais eficiente possível é preciso ter alguns cuidados no que toca à criação da banda, nomeadamente ao nome que se dá em certas características (*features)* bem como a sua organização na árvore do produto. Para melhor compreensão prosseguiremos com um exemplo simples: ao ficheiro *CAD* que contém a banda deu-se o nome de "*Strip_Layout*", sendo que dentro deste está uma subsecção onde foram introduzidos os *extrudes* que dão forma à banda, bem como os *sketchs* que servem de guiamento ao *extrude*. Para mais tarde, durante a programação, podermos proceder à distinção entre estes dois procedimentos, o *sketch* é finalizado por ".S" tal como mostra na imagem 4-3 b). É de notar que o nome atribuído a cada

Extrude/*Sketch* está completamente ao critério do projetista, desde que este siga as instruções de dar o mesmo nome com a diferença de acrescentar ".S" ao *sketch*.



a) b)

Imagem 4-3 a) Ilustração da árvore de produto da banda em estudo; b) Ilustração da seleção de um sketch na árvore de produto.

Havendo a necessidade de colocar várias peças iniciais e respetivos planificados ao longo do comprimento da banda, para que sirvam como uma guia para a sua construção, são colocados fora da subsecção *Strip Layout*. Estas "Peça posicionada_a.b" e "Planificado posicionado_a.b" estão catalogadas por dois números: a e b. Se a respetiva peça estiver posicionada na zona inferior a variável "a" tomará o valor 1, mas se o valor for 2 então a peça está posicionada na zona superior (sendo que esta zona inferior e superior estão definidas tendo em conta a vista do teto da banda). O "b" é referente ao número de peças que já foram importadas para o *CAD*, ou seja, se for preciso importar 8 peças, o valor varia entre 1 e 8.

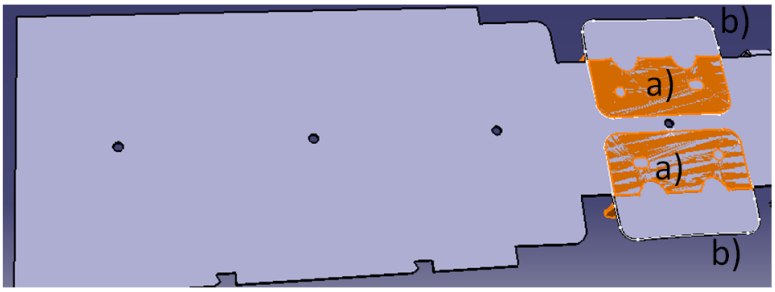


Imagem 4-4 Ilustração das Peças Posicionadas em a) rodeadas pelos Planificados Posicionados em b) respetivos.

Na elaboração da fita foi tido em conta também a redução do desperdício de material. Optou-se então por colocar 2 peças, lado a lado, em cada módulo da banda, ficando estas viradas com as respetivas dobras a 90° para o exterior.

É necessário antes de mais definir os eixos, pois é a partir destes que nasce uma referência comum a toda a ferramenta possibilitando assim uma correta programação/criação dos componentes. Define-se então o eixo das abcissas coincidente com a origem da banda e a meia distância da sua largura, o eixo das ordenadas coincide com a face lateral esquerda da banda e o eixo ZZ coincidente com a face superior ao longo da espessura.

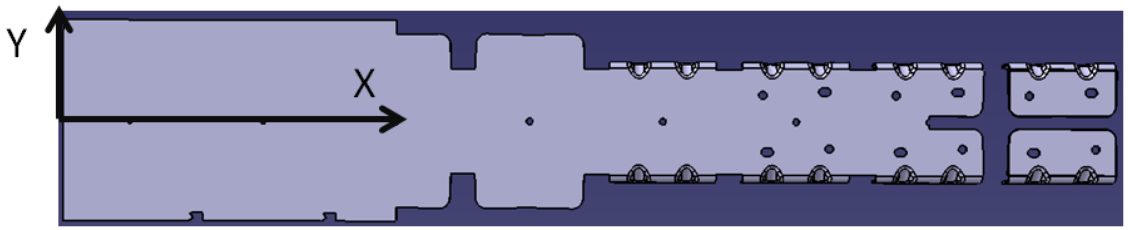


Imagem 4-5 Ilustração da banda final da peça em estudo com o respetivo sistema de coordenadas

4.3.2 Elaboração das macros

Após criação da banda com os respetivos parâmetros impostos inicialmente, pôde-se proceder à elaboração das macros que originam a criação das matrizes e dos punções. Para tal decidiu-se dividir essa tarefa em quatro subtarefas, sendo elas: Matriz de Corte, Punção de Corte, Matriz de Estampar e Punção de Estampar. Logicamente que cada macro vai ter os seus próprios parâmetros e precisará de *inputs* diferenciados. Contudo existem fatores introduzidos pelo projetista que serão comuns a qualquer macro aqui desenvolvida, sendo eles: a espessura da banda, o comprimento da banda, a sua largura bem como a posição relativa a que é colocada, tendo em conta os eixos de coordenadas implementados.

Inicialmente, e ainda sem nos preocuparmos com a programação em si, é preciso importar a banda para o ambiente do *CATIA* v5. A importação será feita para a secção dos *Product*, pois quer-se que o projeto seja constituído por diferentes componentes para se ter liberdade na alteração de algumas peças sem que as outras sofram qualquer modificação. Para tal, qualquer projeto será procedido da respetiva criação do *Product* e importação da respetiva banda. A imagem 4-6 ilustra o procedimento adequado à importação de um componente *3D* para o ambiente do *CATIA*.

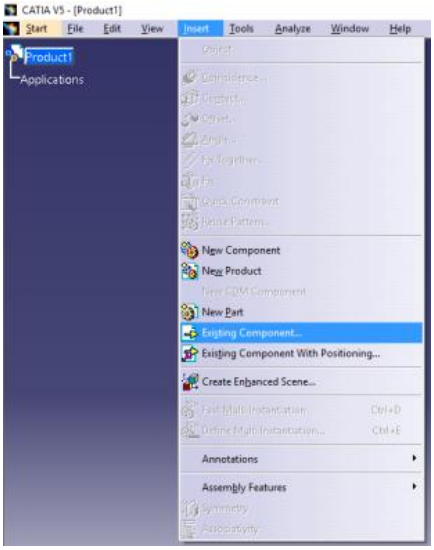
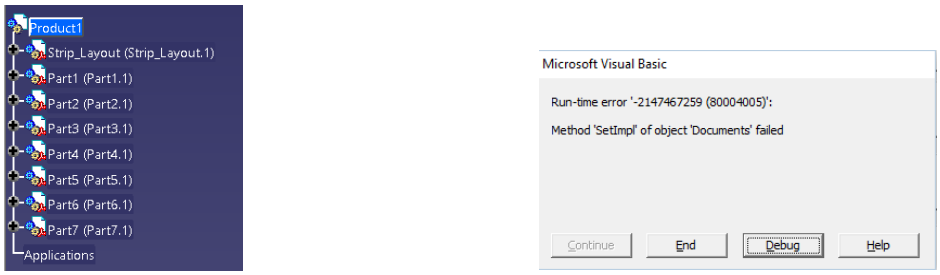


Imagem 4-6 Ilustração da importação de um componente para o ambiente do *CATIA*.

Após esta etapa estar concluída, podemos dar início à criação das respetivas *macros*. É de destacar que a ordem de elaboração das *macros* está ao critério do programador, contudo começou-se por projetar desde a banda até ao exterior. Esta metodologia foi escolhida uma vez que os componentes interiores da ferramenta vão sempre servindo de parâmetros à construção dos componentes exteriores da mesma. É mais fácil projetar os punções depois de ter-mos a banda definida, bem como a criação dos porta-punções após os punções estarem projetados.

4.3.3 Matriz de Corte

Para a criação de uma qualquer *macro* existe a necessidade de questionar o utilizador em qual *Part* quer editar. Este *input* é necessário uma vez que é referente ao número de peças que já fazem parte do conjunto "*Product1*". Poder-se-ia automatizar este parâmetro com o auxílio de um contador que iniciaria em 1 e que seria invocado sempre que uma nova peça seria criada, por exemplo: "nº_da_Part_a_editar = nº_da_Part_a_editar + 1". Deste modo a variável tomaria valores inteiros de 1 até nº de peças totais, sempre de forma incremental. Este pequeno detalhe é importante no *CATIA*, uma vez que o programa necessita que os componentes criados tenham uma sequência crescente, tal como mostra a imagem 4-7 a), e se tal não acontecer é invocada uma mensagem de erro na criação da peça (imagem 4-7 b)).



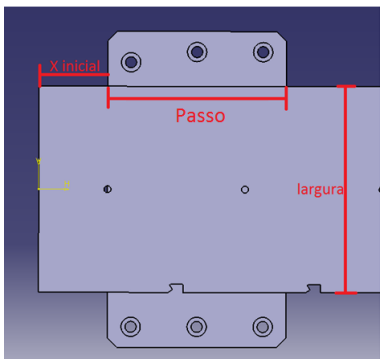
a) b)

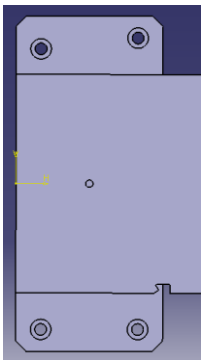
Imagem 4-7 a)Ilustração das *Parts* numeradas de forma incremental; b)Ilustração de um exemplo de uma mensagem de erro no *CATIA*.

Contudo, optou-se por não se proceder à criação dessa rotina, sendo preciso dar a conhecer ao programa, qual o número do componente que está a ser criado, sendo que no exemplo anterior, e posterior invocação de uma nova macro, seria o número 8. Deste modo o utilizador possui uma maior liberdade para introduzir novas peças no produto manualmente, sem necessitar de recorrer a qualquer *macro* criada. Caso contrario e dando o exemplo de querermos criar um componente manualmente este seria o número 8, fazendo com que a próxima *macro* a ser invocada teria de construir a nona peça. Contudo, como manualmente não se consegue incrementar uma unidade ao contador, este possuiria ainda o numero 8, entrando em conflito com o componente já então existente.

Visto que a criação de matrizes é feita uma a uma, e numa ferramenta progressiva existem várias matrizes de corte, então um dos parâmetros fundamentais para a sua elaboração será o seu x inicial, o seu passo e a largura da banda respetiva durante esse mesmo passo. O x inicial é a distância, segundo o eixo das abcissas, a que o projetista quer colocar o início da sua matriz. Esta distância, como todas as outras, está definida em milímetros. Esta será de zero na primeira matriz a ser projetada, isto se o sistema de eixos estiver posicionado segundo a imagem 5-4. Contudo, após esta, o valor será sempre a soma dos passos das matrizes anteriormente projetadas.

Para uma correta construção dos componentes de uma ferramenta, a empresa MCG possui um documento, chamado Caderno de Encargos, onde estão algumas especificações que o projetista deve seguir a quando da elaboração manual do seu projeto. Entre muitas especificações, e falando concretamente da construção de matrizes de corte, está o descentramento de um dos furos de aperto, tal como mostra a imagem 4-8. Não havendo regra empírica para esta característica, o descentramento ocorreu no furo superior esquerdo, como poderia ter acontecido em qualquer um dos outros cinco, ilustrada na imagem 4-8 a).





a) b)

Imagem 4-8 a) Ilustração das dimensões *x inicial*, passo e largura das matrizes; b) Ilustração de uma matriz com passo menor de 100 milímetros.

A espessura que a matriz de corte adquire é outra das especificações que vem no caderno de encargos. Esta tem uma relação direta com a espessura da banda. Visto que a espessura do objeto de estudo é de apenas 1,5 milímetros apenas se procedeu à parametrização para dois exemplos: espessuras de banda inferiores a 1 milímetro ou superiores a este. Para tal recorreu-se à utilização de um ciclo "*if…then…else…*", muito comum no tipo de linguagem *Visual Basic*.

```
If espessura < 1 Then
length5.Value = 25
Else
length5.Value = 35
End If
'
```

Imagem 4-9 Exemplo de um ciclo "*if*" presente na programação da matriz.

Este ciclo compara o valor que o utilizador fornece para o valor da espessura da banda e compara-o com o valor 1. Se este for inferior, então o *extrude* criado para a espessura da matriz possuirá o valor 25, caso contrário a matriz medirá 35 milímetros ao longo do eixo Z. Existe ainda o pormenor de que só são criados 6 furos de aperto se o valor do passo que damos a essa matriz for superior a certos valores, tal como é demostrado na imagem 4-8 a).

A distância da fita ao exterior da matriz segundo o eixo das ordenadas também se encontra catalogada, sendo esta dimensão 10 vezes a espessura da chapa. Na imagem 4-10 está representada um exemplo de bandas iguais, com respetivas matrizes, mas com espessuras diferentes, sendo que a matriz representada a laranja corresponde a uma fita de espessura 0,9 milímetros. É de notar que a localização dos furos também sofre alteração, como não poderia deixar de ser.

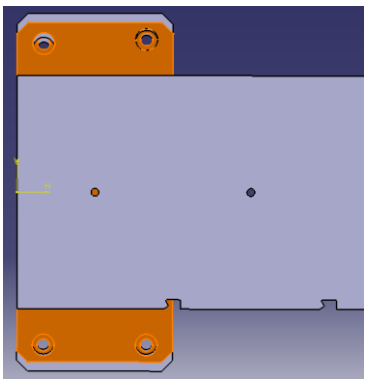
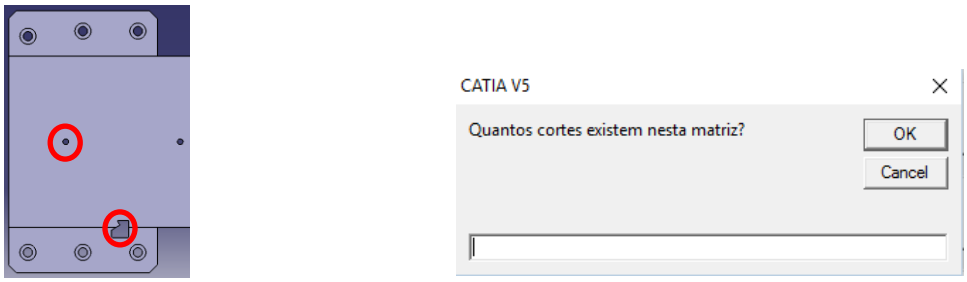


Imagem 4-10 Ilustração de uma matriz com menor (representada a laranja) ou maior largura.

Outra peculiaridade desta *macro* é a possibilidade da criação dos furos por onde o retalho vai sair. Após a introdução dos *inputs* necessários para a criação das matrizes aparecerá uma janela ao utilizador que o questionará sobre o número de cortes existentes na matriz criada, bem como a designação desses mesmos furos. O nome introduzido nesta janela terá de ser o mesmo do nome dado ao *sketch* onde se localiza o contorno do furo. Vamos tomar como exemplo os seguintes *inputs*: espessura de 1,5 mm; passo de 130 mm e largura de banda de 150 mm. Neste caso existem dois punções a atuar, o que implica a necessidade de existência de dois furos por onde sairão o retalho, ilustrados na imagem 4-11 a). Na imagem 4-11 b) está representado o menu onde o projetista introduzirá a quantidade de punções a atuar.



a) b)

Imagem 4-11 a) Ilustração da quantidade de furos existentes na matriz com as especificações do exemplo; b)Ilustração do menu apresentado para obtenção do valor de um parâmetro referente a um *sketch*.

Esta *macro*, apesar de só poder ser invocada quando a largura da matriz for superior à largura da fita, é considerada universal. Tal característica deve-se ao facto de conseguir criar qualquer matriz de corte independentemente da banda em que se está a trabalhar. Não acontecendo o mesmo, por exemplo, na Matriz de Estampar.

4.3.4 Matriz de Estampar

Como foi dito anteriormente, este tipo de macro não é de uso generalizado, sendo só possível a sua atuação nesta mesma banda, ou bandas com características semelhantes a esta, nomeadamente maior, ou menor aproximação das saliências interiores, mas nunca na sua inexistência.

Foi na criação desta macro que surgiram as maiores dificuldades existentes neste projeto. Tal deveu-se à necessidade da criação de uma forma negativa relativa ao formato da peça a estampar que pudesse dar forma ao punção. Para contornar esta dificuldade procedeu-se à subtração da peça na matriz inicial.

Contudo foi preciso eliminar o excesso de material, recorrendo-se para isso à opção "*Remove Face*" do *CATIA*. Como mostra a imagem 4-12 foi necessário proceder à seleção face a face para posterior eliminação das mesmas. É devido a esta seleção face a face que a *macro* criada só dará para este tipo de peça, pois na execução do código de programação, o *CATIA* tentará encontrar estas mesmas faces para posterior eliminação. Como tal não acontecerá em outro tipo de banda, aparecerá uma mensagem de erro muito semelhante à da imagem 4-7 b).

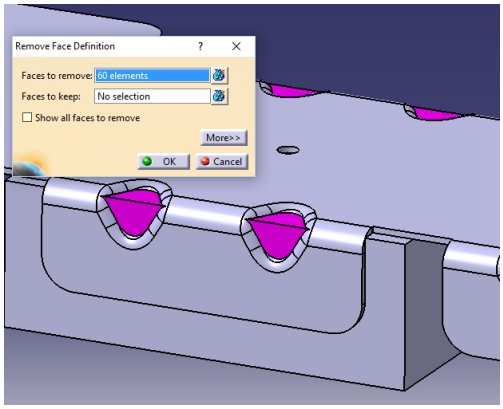


Imagem 4-12 Ilustração da selecção das faces a serem eliminadas pelo *CATIA*.

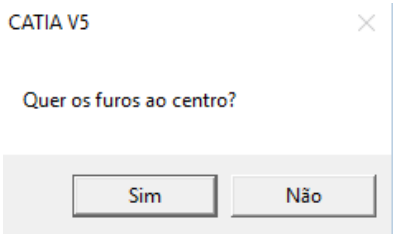
Poder-se-ia subtrair esta secção da linha de código e o utilizador teria, após a execução da *macro*, eliminar as faces que fossem excedentárias, contudo, e como foi escolhida esta mesma peça para a maior automatização possível do projeto da respetiva ferramenta progressiva, não se optou por essa via, reduzindo assim ainda mais o tempo de projeto deste objeto em estudo.

Esta *macro* tem a particularidade de os furos para aperto da matriz ao porta-matrizes poderem ser centralizados, ou descentralizados. Isto foi consequência dos mesmos não poderem estar centrados na operação onde se fariam os furos existentes no centro da peça. Para tal distinção recorreu-se mais

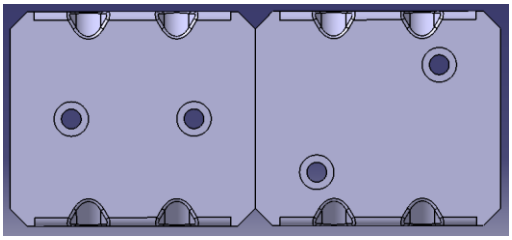
uma vez ao ciclo "*if…then…else…*" onde é questionado ao utilizador se pretende que os furos sejam ao centro. Caso a resposta seja afirmativa a *macro* correrá uma determinada linha de código, caso contrário correrá uma outra. Na imagem 4-13 está ilustrada a programação existente nesta macro, a janela que aparecerá ao projetista bem como os dois resultados possíveis.

```
Dim point2D5 As Point2D
If PERGUNTA2 = vbYes Then
    Set point2D5 = factory2D1.CreatePoint(xinicial + passo - passo / 4, 0) 'centro circunferencia 1
Else
    Set point2D5 = factory2D1.CreatePoint(xinicial + passo - passo / 4, D / 4) 'centro circunferencia 1 descentrado
End If
```

a)



b)



c)

Imagem 4-13 a) Ilustração das linhas de código utilizadas para fixação de um dos centros dos furos nas matrizes de estampar, respetivo menu b) e resultado final c).

Esta *macro* não se encontra cem por cento automatizada. No início é necessário que o utilizador copie e cole a "peça posicionada" respetiva à secção onde se quer criar a matriz de estampagem. Isto deve-se ao facto de essa colocação não ser um simples "*Copy & Paste*", mas sim um "*Paste Special As Result*" não sendo parametrizável no *CATIA*, pelo menos ao nível da gravação de *macros*. A imagem 4-14 representa o menu existente para a colagem por esta via no *CATIA*.

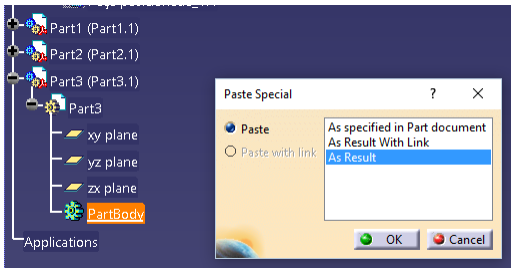


Imagem 4-14 Ilustração do menu "*Paste Special As Result*" no *CATIA*

4.3.5 Punções

A criação dos punções é realizada com base na metodologia utilizada na construção das matrizes sendo que se dividem em dois tipos: Punções de Estampar e Punções de Corte. A *macro* "Punções de Estampar" possui *inputs* muito semelhante aos necessários na *macro* "Matriz de Estampar" no que toca à cópia e colagem ("*Paste Special As a Result*") necessárias antes da sua execução. Para tal é precedida de uma caixa de texto com as instruções que o projetista terá de realizar para a sua correta utilização. Optou-se por dividir esta tarefa em duas *macros* distintas, sendo uma que realiza o punção correspondente à zona superior da banda, e outra que faz a mesma tarefa mas para a zona inferior.

Esta divisão deveu-se ao facto de na linha de código implementada ser necessário recorrer a coordenadas dos respetivos punções. Estas são iguais no que toca ao eixo das abcissas, mas simétricas quando falamos do eixo das ordenadas. Embora pudesse-mos implementar tudo numa só macro, optou-se por não o fazer uma vez que esta possuiria uma linha de código demasiado extensa o que levaria a um aumento da probabilidade de erro a quando da sua elaboração.

No projeto de um punção de dobra um dos fatores que teremos de ter em conta é a sua largura, uma vez que este terá dimensões mínimas mediante a sua resistência à flexão. Pela tabela 4-1 denotamos na existência de uma fórmula para cálculo da flexa máxima neste tipo de caso.

Tabela 4-1 Ilustração da flecha máxima numa viga encastrada.

Viga	Carregamento (comprimento L)	Deflexão angular na extremidade +	Flecha Máxima + ↑
1		$\phi = -PL^2 / 2EI$	$f = -PL^3 / 3EI$

A força P está diretamente relacionada com a capacidade da prensa, visto que é o único fator responsável para a carga descendente (note-se que se optou por desprezar o peso dos componentes que constituem a parte superior da ferramenta progressiva, visto que estes ainda não se encontram desenvolvidos nesta fase de projeto). A variável onde é traduzida a largura do punção é no momento de inércia (I) que será indicado pela fórmula: $I = \frac{hb^3}{12}$, onde b é a largura e h o comprimento da secção retangular do punção.

Deste modo a equação retirada da Tabela 4-1 ficará:

$$f = \frac{P.L^3}{3.E\frac{h.b^3}{12}} = \frac{12.P.L^3}{3.E.h.b^3} \qquad \text{(equação 1)}$$

Querendo que por esta equação se obtenha a largura do punção teremos de dar valores a todas as outras variáveis, sendo que se admitiu um valor de flecha máximo de 0,5 milímetros.

Rearranjando a equação 1 ficamos com:

$$b = \sqrt[3]{\frac{12.P.L^3}{3.E.h.f}} \qquad \text{(equação 2)}$$

Onde:

$P = carga\ exercida\ no\ punção$

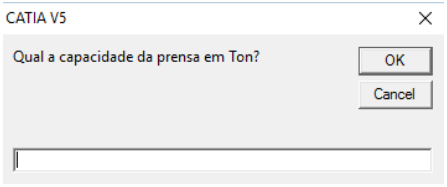
$L = semi - altura\ do\ punção$

$E = Modulo\ de\ Elasficidade\ do\ Punção$

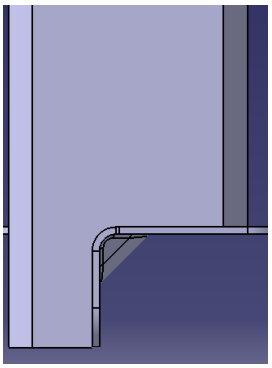
$h = comprimento\ do\ punção$

$f = flexa\ máxima\ permitida$

Optou-se por programar a macro só para esta peça em concreto, mas feita em prensas com capacidades diferentes, ou seja, da equação anterior o utilizador apenas terá de introduzir o *input* referente à força P, sendo que as restantes variáveis já estão pré-estabelecidas na macro. Pela imagem 4-15 observa-se o menu que aparece ao utilizador, bem como o punção devidamente desenvolvido.



a)



b)

Imagem 4-15 a) Ilustração do menu referente à capacidade da prensa; b)Ilustração da vista de perfil do punção de estampar.

Relativamente aos punções de corte, existem três vias distintas, correspondentes a três *macros* distintas, para a sua elaboração denominadas: “Piloto”, “Punções Standard” e “Punção Especifico 1”.

A *macro* “Piloto” foi desenvolvida para receber as coordenadas relativas ao centro geométrico de um furo existente na banda onde se queira a ação de um punção. Este punção possui sempre uma forma circular com uma cabeça cilíndrica, sendo que esta varia o seu comprimento mediante a altura total do punção. Pode ser invocada sempre que haja a necessidade de realização de um corte circular simples descendente, independentemente da fita em que se está a trabalhar.

Para uma maior abrangência no desenvolvimento dos punções de corte optou-se também por realizar uma *macro* que projete qualquer formato de punção desde que este esteja previamente concebido na banda. Estes formatos são os mesmos utilizados na realização dos furos para a saída do retalho na construção das matrizes. Uma particularidade deste tipo de *macro* é a necessidade de o projetista ter de localizar os furos para posterior aperto ao porta-punções num *sketch* diferente deste. Apesar desta via possuir maior abrangência na forma dos punções, pode também possuir maior erro na sua realização, pois requer maior grau de cuidado na realização da fita.

A *macro* “Punção Específico 1” é em tudo semelhante à *macro* “Piloto”, diferenciando-se desta somente na forma do punção. Este possui o contorno do furo representado na figura 4-2 (b). Os *inputs* necessários são a coordenada de um ponto específico bem como a altura do punção.

Após a realização das *macros* para projeto das matrizes e respetivos punções, obtém-se já um pequeno esboço da ferramenta progressiva. É de notar que as *macros* criadas a partir daqui são em tudo semelhantes a estas, criando peças que podem ser editáveis em qualquer altura do projeto, estando a critério do projetista. Visto que as mesmas não possuem ligação direta entre a banda, sempre que for preciso alterar algo na fita, como dimensões ou espessuras, poderá ser preciso apagar uma determinada peça e voltar a construi-la recorrendo novamente à *macro* respetiva. Este processo poderá tornar-se demorado se a banda não estiver corretamente construída. A imagem 4-16 demostra o resultado final da utilização destas *macros* faladas anteriormente.

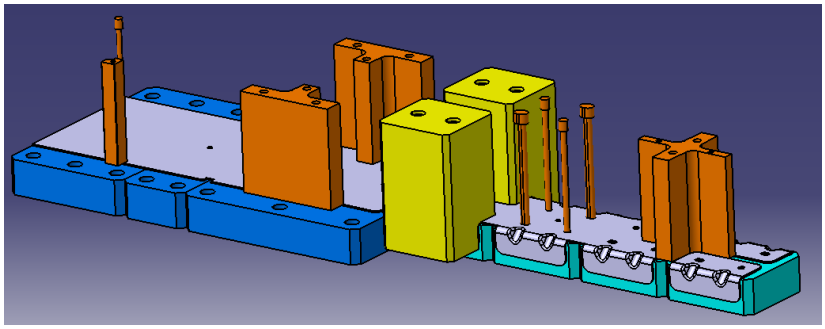


Imagem 4-16 Ilustração dos punções e respetivas matrizes na banda.

4.3.6 Sub-Base

Ao contrário do que tem sido desenvolvido até aqui, peças individuais dos componentes da ferramenta progressiva, a *macro* "Sub-Base" cria não uma, mas um conjunto de componentes que engloba a sub-base propriamente dita, os casquilhos, que farão o guiamento do movimento ascendente e descendente da ferramenta, os aperta-casquilhos e por último os batentes. Para que este conjunto de componentes fique corretamente desenvolvido é necessário dar os seguintes *inputs*:

- ➔ Comprimento total da banda
- ➔ Largura máxima da banda
- ➔ Espessura da fita
- ➔ Altura dos batentes

Os casquilhos são um componente muito importante para o bom uso da ferramenta progressiva e para o aumento da sua vida útil. Eles ajudam no guiamento da mesma evitando assim deformações na peça final obtida e consequentemente nos esforços criados pela prensa.

Esta *macro* não leva em conta o formato da banda que está a ser desenvolvida, uma vez que foi programada com o auxílio do caderno de encargos da empresa. A espessura da banda é necessária para que a base possa ter uma coordenada fixa no eixo dos ZZ, uma vez que quanto maior a espessura, mais inferior será o inicio da base. Com a criação desta *macro*, apenas faltaria elaborar o resto do corpo inferior da ferramenta progressiva, destacando-se os cutelos, a base propriamente dita e a prensa inferior, que ostentará a própria ferramenta progressiva. Note-se na existência de uma pequena rampa, após o punção que faz a separação, para ajudar a direcionar as peças produzidas. O retraço é todo direcionado para a abertura circular existente na base. Assim o componente final nunca entra em contacto com o material excedentário.

A base é o elemento de ligação entre a ferramenta e a prensa, estando a sua dimensão limitada pelo tamanho da banda e da mesa da prensa. É necessário garantir que existem pontos de fixação neste elemento coincidente com a mesa da prensa. Existem ainda quatro reentrâncias laterais que tem como objetivo a correta deslocação e montagem da ferramenta na prensa, sendo para isso necessária uma grua própria existente na empresa MCG. A base acoplada aos elementos representados na imagem 4-16 fica com o aspeto observado na imagem 4-17.

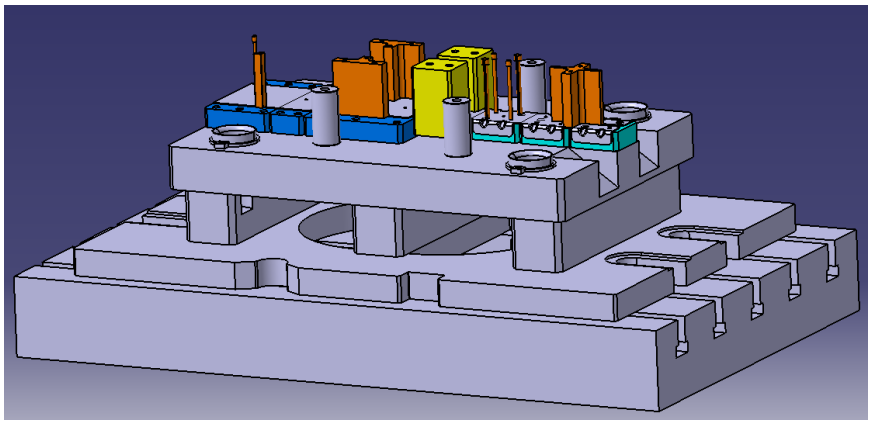


Imagem 4-17 Ilustração da parte inferior desenvolvida da ferramenta progressiva.

4.3.7 Zona Superior

Da parte superior da ferramenta progressiva fazem parte o pisa, o porta-pisa, o porta-punções, a placa de choque e o teto propriamente dito. Para que não haja deformação enquanto os punções estão a atuar existe uma peça fundamental na ferramenta progressiva com o nome de pisa, responsável por este efeito. Este componente está acoplado ao porta-pisa que por sua vez se desloca solidariamente ao teto.

Para a localização do porta-punções há a necessidade de conhecer 3 coordenadas fundamentais: o x inicial, y inicial e o passo. A partir destes três *inputs* o projetista pode posicionar o porta-punções onde bem entender, criar um para cada punção, ou no limite, criar um único porta-punções que englobe os punções todos. Um bom exemplo das vantagens da criação de um projeto parametrizável é na elaboração da placa de choque, responsável pela minimização do impacto criado na atuação dos punções. Este componente é muito semelhante ao porta-punções, diferenciando-se deste em pequenos detalhes tais como a sua espessura e a sua localização ao longo do eixo Z. Contudo, a sua localização, segundo o plano XY, na ferramenta progressiva tais como os seus *inputs* necessários, são exatamente os mesmos, o que possibilita a sua elaboração ao mesmo tempo do porta-punções, estando este uma coordenada inferior no eixo dos ZZ.

Todas as dimensões de espessuras de componentes foram obtidas com a ajuda do caderno de encargos da empresa. Embora a ferramenta progressiva não fique cem por cento finalizada com a elaboração destas *macros*, fica muito próxima disso, faltando apenas a localização de pequenos furos para aperto de componentes. Essa localização deixou-se ao critério do projetista, pois por vezes não possuem forma *standard*, seja para economia de espaço ou simplesmente por experiência do engenheiro, sabendo este o melhor sítio para a sua fixação.

Na imagem 4-18 podemos observar os vários constituintes da ferramenta projetada. A cinza destacam-se os componentes desenvolvidos através das *macros* mencionadas nos subcapítulos anteriores. O componente a amarelo é uma possível solução para o porta-punções estando logo de seguida a placa-choque representada a encarnado.

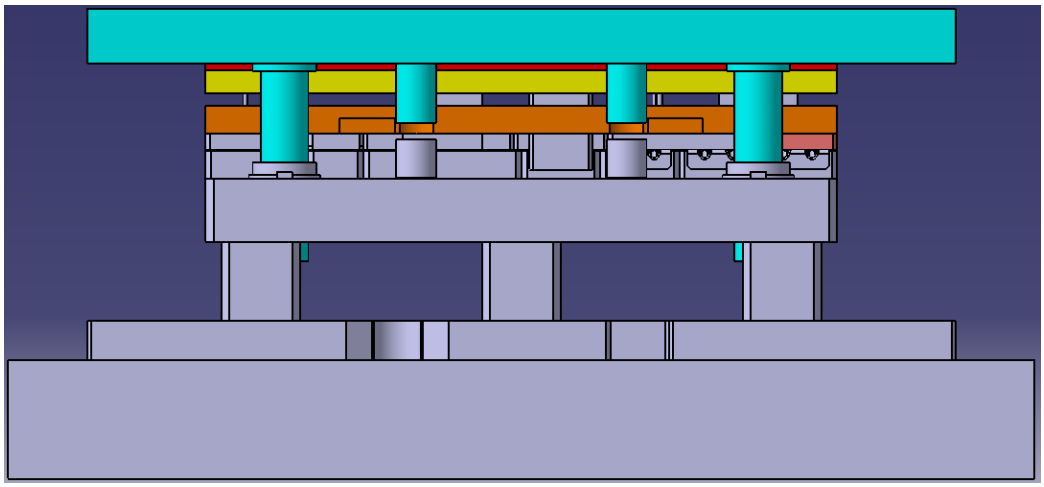


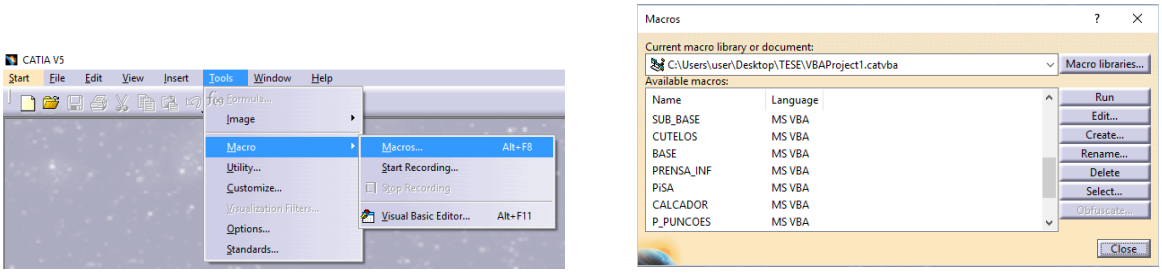
Imagem 4-18 Ilustração da vista em alçado principal da ferramenta produzida.

4.4 Aspeto visual

Nos anos que deram inicio à programação, o aspeto visual dos programas desenvolvidos era essencialmente texto, não se atribuindo muita importância a este fator. Contudo, nos dias correntes o cenário mudou drasticamente e a forma como um programa é mostrado nos monitores tem muita influência, estando cientificamente provado que o cérebro processa informação visual 60 mil vezes mais rapidamente do que informação em texto.

Devido a este fator optou-se por criar uma interface utilizador/programa que fosse minimamente agradável, por um lado, e por outro que resumisse a informação necessária a quando da elaboração da ferramenta progressiva.

O mecanismo básico para que o utilizador possa criar uma peça recorrendo a uma *macro* previamente elaborada é proceder como mostra imagem 4-19 a), ou carregando no atalho no teclado "ALT+F8", tendo de seguida escolher o tipo de *macro* que quer utilizar (imagem 4-19 b)).



a) b)

Imagem 4-19 a) Ilustração da sequência de comandos a seguir para abertura do menu das *macros*; b) Ilustração do menu das *macros* do *CATIA.*

Este processo pode-se tornar demorado e enfadonho, visto que podem estar criadas centenas de *macros* e o utilizador tem de procurar aquela que melhor se adequa ao projeto. Outro aspeto negativo que se destaca é o facto das *macros* se encontrarem divididas individualmente. Dando um exemplo para melhor perceção: das *macros* visualizadas na imagem 4-19 b), as cinco primeiras têm pelo menos um *input* comum, nomeadamente o tamanho total da banda. O facto de utilizarmos as macros por via tradicional implicaria a introdução deste *input* 5 vezes, estando por um lado a minimizar o tempo de projeto de uma ferramenta progressiva com a realização de macros, mas por outro a aumenta-lo ligeiramente com um processo repetitivo e desnecessário.

Para contornar esse problema levantou-se os principais parâmetros que serão comuns a várias linhas de programação, nomeadamente:

→Comprimento da banda

→Largura da banda

→$x\ inicial$

→Espessura da banda

Com isto procedeu-se à criação de um menu mais intuitivo ao utilizador, estando ele representado na imagem 4-20.

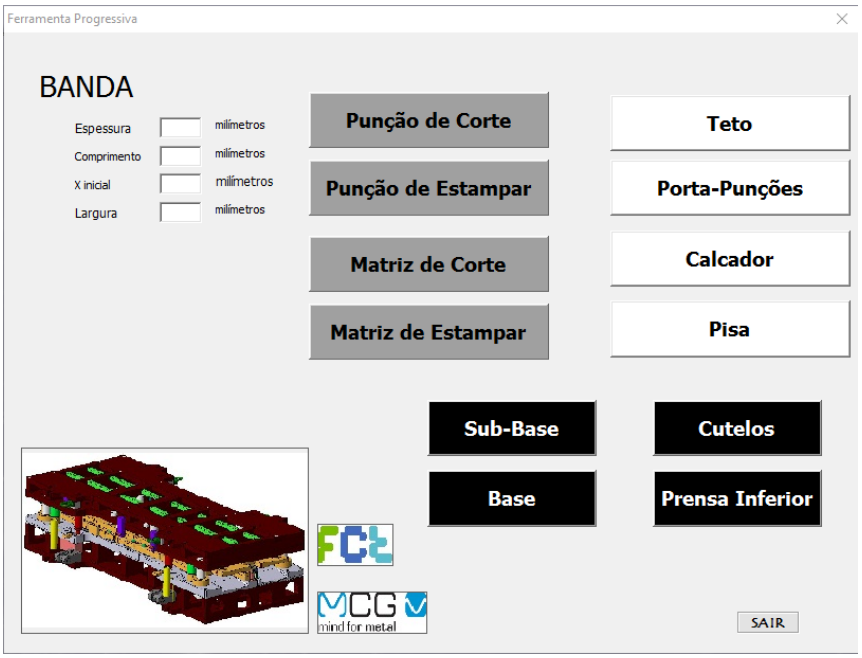


Imagem 4-20 Ilustração da interface criada no *CATIA* para invocação das *macros*.

Dividiu-se a interface criada em 3 zonas distintas. Os botões representados a cinzento representam componentes diretamente desenvolvidos a partir da banda, sendo nestes a existência de cuidados especiais a quando da elaboração da fita. Os botões brancos fazem parte da zona superior da ferramenta progressiva, sendo que os botões de cor negra representam a zona inferior da mesma.

5

Discussão dos Resultados

Este trabalho teve como objetivo primordial a redução do tempo de um projeto de uma ferramenta progressiva através da automatização do mesmo utilizando o *CATIA v5*. Tal como foi descrito nos capítulos anteriores apenas se automatizou a ferramenta para uma peça-exemplo. Com o desenvolvimento da dissertação, não só foi automatizado o projeto de uma ferramenta progressiva, como também criada uma nova metodologia de trabalho explorando as potencialidades do *software* presente na empresa. Após a criação de um novo projeto utilizando esta nova metodologia é imprescindível que seja efetuada uma revisão crítica do mesmo por um projetista com experiência, detetando erros ou falhas que possam ser criadas pela aplicação.

A validação desta dissertação baseia-se na comparação dos tempos de projeto anteriormente obtidos, com os atuais após ser implementadas as metodologias propostas, demostrando assim a possibilidade do aumento do número de projetos a desenvolver pela empresa. Para efeitos de quantificação dos ganhos em termos de tempo definiram-se duas subfases dentro da fase de projeto de uma ferramenta progressiva. A primeira denomina-se por "Elementos estruturais" e engloba o projeto de todos os elementos que irão realizar trabalho nas várias operações de corte bem como dos elementos que servirão de estrutura a todas as operações da ferramenta. Na segunda fase "Elementos periféricos" são contemplados todos os outros elementos que mesmo não realizando trabalho nas várias operações, são indispensáveis ao correto funcionamento da ferramenta (elevadores, extratores, parafusos, cavilhas), assim como a correção de pormenores também importantes tais como colisões entre elementos ou folgas funcionais.

Os dados apresentados na tabela 5-1 são ilustrativos da redução de tempo que esta metodologia pode proporcionar. Foram obtidos contabilizando o tempo que se demorou no projeto de uma ferramenta progressiva por uma e outra via.

Tabela 5-1 Comparação dos tempos pelos dois métodos estudados

	Método "Tradicional"	**Método "Otimizado"**	**Redução**
Elaboração da Banda	10 horas	13 horas	-33%
Projeto da ferramenta	56 horas	40 horas	29%
Elementos estruturais	40 horas	24 horas	40%
Elementos periféricos	16 horas	16 horas	0%
TOTAL	66 horas	53 horas	20%

Sendo o método "Tradicional" o nome dado à antiga metodologia existente na empresa e o método "Otimizado" a nomenclatura atribuída à nova metodologia imposta (após a criação das macros), prova-se assim que esta metodologia do projeto aplicada a todas as ferramentas progressivas pode constituir um ganho significativo para a empresa. O processo de criação e desenvolvimento da ferramenta progressiva teve a duração aproximada de 4 meses, sendo o primeiro mês de formação básica do *software* utilizado. Destaca-se ainda que todo o desenvolvimento deste projeto bem como a recolha do tempo de comparação presente na tabela 5-1 foram realizados por um utilizador sem experiência nenhuma na utilização do *CATIA*, podendo os resultados terem sido diferentes se realizados por um projetista com prática e destreza na programação.

Com isto, a elaboração desta dissertação revelou que ainda existe um longo caminho para que o projeto de uma ferramenta progressiva consiga ser 100% automatizado. Neste caso em particular, embora se consiga reduzir o tempo, sobretudo no projeto dos elementos estruturais de uma ferramenta progressiva, não o foi possível fazer sem termos tido a necessidade de aumentar o tempo de elaboração da banda em cerca de 30%. Este aumento deveu-se às necessidades de alterações na fita, que foram criadas com o objetivo do bom funcionamento das *macros*. Com um engenheiro especialista em programação e com bons conhecimentos do CATIA, estas adversidades poderiam ter sido ultrapassadas. No limite não se despenderia mais nenhuma hora do que as 10 necessárias para a elaboração da banda. As *macros* geradas nesta dissertação também são suscetíveis de melhoramento, sendo possível reduzir o número de *inputs* necessários atualmente, bem como criada uma dependência constante da banda, fazendo com que alterações na mesma proporcionem as devidas alterações nos componentes respetivos.

6

Conclusões

No capítulo anterior desta Dissertação Final de Mestrado foi apresentado um método designado como "Método Otimizado" que delineou uma estratégia de otimização da modelação de um projeto *3D*. Com esse método provou-se a poupança de cerca de 20% relativamente ao método existente na MCG.

Na fase de projeto da ferramenta tornou-se evidente que o método "Otimizado" não permitiria a elaboração na íntegra do projeto da ferramenta, sendo sempre necessária alguma modelação 3D adicional após aplicadas as *macros*. Esta necessidade é justificada pela especificidade que cada projeto apresenta e que cada empresa de construção de ferramentas apresenta.

Embora o método "Otimizado" apresente um aumento do tempo na fase da elaboração da banda, este é inteiramente compensado a quando do projeto da ferramenta propriamente dito. Este aumento deve-se ao maior número de horas que é preciso despender para criar as especificações necessárias ao bom funcionamento das *macros*.

Por este meio conclui-se que foram atingidos todos os objetivos inicialmente propostos:

- Ficou demostrada a minimização dos tempos de projeto de uma ferramenta progressiva em cerca de 20%
- Com esta redução pode afirmar-se que existe uma redução dos custos totais do projeto, mesmo que estes só consigam ser atingidos a longo prazo
- A redução do tempo potencia também um aumento da qualidade do serviço, reduzindo o prazo de entrega do produto final

No seguimento desta dissertação é sugerido o aperfeiçoamento da automatização e consequente melhoramento deste projeto, otimizando os tempos de projeto de elementos com características semelhantes.

Bibliografia

[1] IndiaCadWorks, "Catia: A CAD Software Review," 15 Maio 2013. [Online]. Available: http://www.indiacadworks.com/blog/catia-a-cad-software-review/. [Acedido em Julho 2016].

[2] CENIT, "CATIA - The allround fenius for product development," CENIT, [Online]. Available: http://www.cenit.com/en_EN/plm/3ds-plm/software/catia.html. [Acedido em Julho 2016].

[3] Dassault Systèmes, "CATIA Ingrastructure," Dassault Systèmes, Maio 2015. [Online]. Available: http://maruf.ca/files/catiahelp/CATIA_P3_default.htm. [Acedido em Julho 2016].

[4] tech-clarity, "Siemens NX Cad vision 2014+," 16 Abril 2014. [Online]. Available: http://tech-clarity.com/nx-cad-vision/3744. [Acedido em Julho 2016].

[5] Siemens, "Teamcenter," Siemens, [Online]. Available: http://www.plm.automation.siemens.com/en_us/products/teamcenter/design-datamanagement/index.shtml. [Acedido em Julho 2016].

[6] Siemens, *NX programming and customization,* Brochura Siemens, 2015.

[7] Siemens, *NX Tooling,* Brochura Siemens, 2014.

[8] Dassault Systemes, "Compay History," SolidWorks. [Online]. Available: https://www.solidworks.com/sw/656_ENU_HTML.htm. [Acedido em Julho 2016].

[9] Accurate Die Design, "The History of Logopress," 2014. [Online]. Available: http://www.accuratediedesign.com/logopresshistory.php. [Acedido em Julho 2016].

[10] Logopress3, "Unbending Functions," Logopress3, [Online]. Available: http://www.logopress3.com/en/products-1.php. [Acedido em Julho 2016].

[11] Logopress3, "Logopress experience the power," Logopress3, [Online]. Available: http://www.logopress3.com/en/products-2.php. [Acedido em Julho 2016].

[12] Siemens, [Online]. Available: www.plm.automation.siemens.com. [Acedido em Julho 2016].

[13] Diogo F. N. Morgado, "Optimização do processo de projecto de uma ferramenta progressiva para a estampagem de componentes metálicos", *Instituto Superior de Engenharia de Lisboa,* 2013

[14] João M. B. C. M. Faria, “Ferramentas Progressivas Híbridas – Estudo de caso”, Universidade do Minho, 2013

[15] Ricardo A. M. N. Manso, “Optimização do desenvolvimento do projecto de ferramenta progressiva”, Faculdade de Ciências e Tecnologias da Universidade Nova de Lisboa, 2015

[16] A. Borges, “Manual para Construção de Ferramentas Progressivas”, Manuel da Conceição Graça, 2009

[17] Apêndice D Flechas Vigas2.pdf, [Online].
Available: http://www.uff.br/petmec/downloads/resmat/U-ApendiceDFlechasVigas2.pdf
[Acedido em Agosto 2016]

Anexo A

Programação da matriz de corte

```
'
Sub CATMain()                                                   '----------> Ínicio do programa
'
Dim productDocument1 As ProductDocument
Set productDocument1 = CATIA.ActiveDocument
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Dim NUM As Integer                               '''''''''''''''''''----------> Exemplo de como declarar uma variável e posterior interação com o utilizador
NUM = InputBox("Em que Part. quer editar?")      '''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
Dim product1 As Product
Set product1 = productDocument1.Product
'
Dim products1 As Products
Set products1 = product1.Products
'
Dim product2 As Product
Set product2 = products1.AddNewComponent("Part", "")    '--------------------> Adiciona um novo componente ao Conjunto
'
Dim move1 As Move
Set move1 = product2.Move
'
Set move1 = move1.MovableObject
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''
Dim arrayOfVariantOfDouble1(11)              ''''''''''
arrayOfVariantOfDouble1(0) = 1#              ''''''''''
arrayOfVariantOfDouble1(1) = 0#              ''''''''''
arrayOfVariantOfDouble1(2) = 0#              ''''''''''
arrayOfVariantOfDouble1(3) = 0#              ''''''''''
arrayOfVariantOfDouble1(4) = 1#              ''''''''''
arrayOfVariantOfDouble1(5) = 0#              ''''''''''---------> Faz coincidir as coordenadas locais do novo componente com as coordenadas globais
arrayOfVariantOfDouble1(6) = 0#              ''''''''''           da banda já existente
arrayOfVariantOfDouble1(7) = 0#              ''''''''''
arrayOfVariantOfDouble1(8) = 1#              ''''''''''
arrayOfVariantOfDouble1(9) = 0#              ''''''''''


arrayOfVariantOfDouble1(10) = 0#             ''''''''''
arrayOfVariantOfDouble1(11) = 0#             ''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
Set move1Variant = move1
move1Variant.Apply arrayOfVariantOfDouble1
'
Dim documents1 As Documents
Set documents1 = CATIA.Documents
'
Dim partDocument1 As PartDocument
Set partDocument1 = documents1.Item("Part" & NUM & ".CATPart")  '----------> Adiciona uma nova PART. com o número "NUM" que foi declarado anteriormente
'
Dim part1 As Part
Set part1 = partDocument1.Part
'
Dim hybridBodies1 As HybridBodies
Set hybridBodies1 = part1.HybridBodies
'
Dim hybridBody1 As HybridBody
Set hybridBody1 = hybridBodies1.Add()
'
Dim hybridShapeFactory1 As HybridShapeFactory
Set hybridShapeFactory1 = part1.HybridShapeFactory
'
Dim originElements1 As OriginElements
Set originElements1 = part1.OriginElements
'
Dim hybridShapePlaneExplicit1 As HybridShapePlaneExplicit
Set hybridShapePlaneExplicit1 = originElements1.PlaneXY
'
Dim reference1 As Reference
Set reference1 = part1.CreateReferenceFromObject(hybridShapePlaneExplicit1)
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Dim espessura, xinicial, largura, passo As Double      ''''''''''''''''''''''''''''''''''''''''''''''''
espessura = InputBox("Qual a espessura da chapa?")     ''''''''''''''''''''''''''''''''''''''''''''''''------> Declara o resto das variáveis necessárias


xinicial = InputBox("Qual o X inicial?")               ''''''''''''''''''''''''''''''''''''''''''''''''        e questiona o utilizador sobre os
largura = InputBox("Qual a largura da banda?")         ''''''''''''''''''''''''''''''''''''''''''''''''        seus valores
passo = InputBox("Qual o passo?")                      ''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
Dim hybridShapePlaneOffset1 As HybridShapePlaneOffset
Set hybridShapePlaneOffset1 = hybridShapeFactory1.AddNewPlaneOffset(reference1, espessura + 0.75, True) '-----> criação de um plano auxiliar
'
hybridBody1.AppendHybridShape hybridShapePlaneOffset1
'
part1.InWorkObject = hybridShapePlaneOffset1
'
part1.Update
'
Dim bodies1 As Bodies
Set bodies1 = part1.Bodies
'
Dim body1 As Body
Set body1 = bodies1.Item("PartBody")
'
part1.InWorkObject = body1
'
Dim sketches1 As Sketches
Set sketches1 = body1.Sketches
'
Dim hybridShapes1 As HybridShapes
Set hybridShapes1 = hybridBody1.HybridShapes
'
Dim reference2 As Reference
Set reference2 = hybridShapes1.Item("Plane.1")      '----------> Nome dado ao plano auxiliar
'
Dim sketch1 As Sketch
Set sketch1 = sketches1.Add(reference2)
'
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Dim arrayOfVariantOfDouble2(8)                              '''''''''
arrayOfVariantOfDouble2(0) = 0#                             '''''''''
```

```
arrayOfVariantOfDouble2(1) = 0#                           '''''''''
arrayOfVariantOfDouble2(2) = -espessura - 0.75            '''''''''----------> Faz coincidir as coordenadas locais do plano criado com as coordenadas
arrayOfVariantOfDouble2(3) = 1#                           '''''''''            globais (localizadas a 0.75 mm da face superior no sentido positivo) da banda
arrayOfVariantOfDouble2(4) = 0#                           '''''''''            já existente, com um offset igual à espessura da banda + 0.75, uma vez que
arrayOfVariantOfDouble2(5) = 0#                           '''''''''            se quer um plano coincidente com a face inferior da fita
arrayOfVariantOfDouble2(6) = 0#                           '''''''''
arrayOfVariantOfDouble2(7) = 1#                           '''''''''
arrayOfVariantOfDouble2(8) = 0#                           '''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
Set sketch1Variant = sketch1
sketch1Variant.SetAbsoluteAxisData arrayOfVariantOfDouble2
'
part1.InWorkObject = sketch1
'
Dim factory2D1 As Factory2D
Set factory2D1 = sketch1.OpenEdition()
'
Dim geometricElements1 As GeometricElements
Set geometricElements1 = sketch1.GeometricElements
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Dim axis2D1 As Axis2D                                         ''''''''''''''
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")         ''''''''''''''
'                                                             ''''''''''''''
Dim line2D1 As Line2D                                         ''''''''''''''
Set line2D1 = axis2D1.GetItem("HDirection")                   ''''''''''''''----------> Faz coincidir o sistema de eixos
'                                                             ''''''''''''''
line2D1.ReportName = 1                                        ''''''''''''''
'                                                             ''''''''''''''
Dim line2D2 As Line2D                                         ''''''''''''''
Set line2D2 = axis2D1.GetItem("VDirection")                   ''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
line2D2.ReportName = 2
'
'-----------------------> Procede-se agora à criação da matriz, tendo de ser introduzida as coordenadas de cada ponto
'                         e posterior união através de uma linha
'
'
Dim point2D1 As Point2D
Set point2D1 = factory2D1.CreatePoint(xinicial + 7.071068, largura / 1.5 + 10 * espessura) 'ponto 1
'
point2D1.ReportName = 3
'
Dim point2D2 As Point2D
Set point2D2 = factory2D1.CreatePoint(xinicial + passo - 7.071068, largura / 1.5 + 10 * espessura) 'ponto 2
'
point2D2.ReportName = 4
'
Dim line2D3 As Line2D
Set line2D3 = factory2D1.CreateLine(xinicial + 7.071068, largura / 1.5 + 10 * espessura, xinicial + passo - 7.071068, largura / 1.5 + 10 * espessura)   'linha 1
'
line2D3.ReportName = 5
'
line2D3.StartPoint = point2D1
'
line2D3.EndPoint = point2D2
'
Dim point2D3 As Point2D
Set point2D3 = factory2D1.CreatePoint(xinicial + passo, largura / 1.5 + 10 * espessura - 7.071068) 'ponto 3
'
point2D3.ReportName = 6
'
Dim point2D4 As Point2D
Set point2D4 = factory2D1.CreatePoint(xinicial + passo, -largura / 1.5 - 10 * espessura + 7.071068) 'ponto 4
'
point2D4.ReportName = 7
'
Dim line2D4 As Line2D
Set line2D4 = factory2D1.CreateLine(xinicial + passo, largura / 1.5 + 10 * espessura - 7.071068, xinicial + passo, -largura / 1.5 - 10 * espessura + 7.071068)   '
'
line2D4.ReportName = 8
'
line2D4.EndPoint = point2D3
'
line2D4.StartPoint = point2D4
'
Dim point2D5 As Point2D
Set point2D5 = factory2D1.CreatePoint(xinicial + passo - 7.071068, -largura / 1.5 - 10 * espessura) 'ponto 5
'
point2D5.ReportName = 9
'
Dim point2D6 As Point2D
Set point2D6 = factory2D1.CreatePoint(xinicial + 7.071068, -largura / 1.5 - 10 * espessura) 'ponto 6
'
point2D6.ReportName = 10
'
Dim line2D5 As Line2D
Set line2D5 = factory2D1.CreateLine(xinicial + passo - 7.071068, -largura / 1.5 - 10 * espessura, xinicial + 7.071068, -largura / 1.5 - 10 * espessura)   'linha 3
'
line2D5.ReportName = 11
'
line2D5.StartPoint = point2D5
'
line2D5.EndPoint = point2D6
'
Dim point2D7 As Point2D
Set point2D7 = factory2D1.CreatePoint(xinicial, -largura / 1.5 - 10 * espessura + 7.071068) 'ponto 7
'
point2D7.ReportName = 12
'
Dim point2D8 As Point2D
Set point2D8 = factory2D1.CreatePoint(xinicial, largura / 1.5 + 10 * espessura - 7.071068) 'ponto 8
'
point2D8.ReportName = 13
'
Dim line2D6 As Line2D
Set line2D6 = factory2D1.CreateLine(xinicial, -largura / 1.5 - 10 * espessura + 7.071068, xinicial, largura / 1.5 + 10 * espessura - 7.071068)   'linha 4
'
line2D6.ReportName = 14
```

```
'
line2D6.EndPoint = point2D7
'
line2D6.StartPoint = point2D8
'
Dim constraints1 As Constraints
Set constraints1 = sketch1.Constraints
'
Dim reference3 As Reference
Set reference3 = part1.CreateReferenceFromObject(line2D3)
'
Dim reference4 As Reference
Set reference4 = part1.CreateReferenceFromObject(line2D1)
'
Dim constraint1 As Constraint
Set constraint1 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference3, reference4)
'
constraint1.Mode = catCstModeDrivingDimension
'
Dim reference5 As Reference
Set reference5 = part1.CreateReferenceFromObject(line2D5)
'
Dim reference6 As Reference
Set reference6 = part1.CreateReferenceFromObject(line2D1)
'
Dim constraint2 As Constraint
Set constraint2 = constraints1.AddBiEltCst(catCstTypeHorizontality, reference5, reference6)
'
constraint2.Mode = catCstModeDrivingDimension
'
Dim reference7 As Reference
Set reference7 = part1.CreateReferenceFromObject(line2D4)
'
Dim reference8 As Reference
Set reference8 = part1.CreateReferenceFromObject(line2D2)
'
Dim constraint3 As Constraint
Set constraint3 = constraints1.AddBiEltCst(catCstTypeVerticality, reference7, reference8)
'
constraint3.Mode = catCstModeDrivingDimension
'
Dim reference9 As Reference
Set reference9 = part1.CreateReferenceFromObject(line2D6)
'
Dim reference10 As Reference
Set reference10 = part1.CreateReferenceFromObject(line2D2)
'
Dim constraint4 As Constraint
Set constraint4 = constraints1.AddBiEltCst(catCstTypeVerticality, reference9, reference10)
'
constraint4.Mode = catCstModeDrivingDimension
'
Dim line2D7 As Line2D
Set line2D7 = factory2D1.CreateLine(xinicial, largura / 1.5 + 10 * espessura - 7.071068, xinicial + 7.071068, largura / 1.5 + 10 * espessura)    'linha 5
'
line2D7.ReportName = 15
'
line2D7.StartPoint = point2D8
'
line2D7.EndPoint = point2D1
'
Dim reference11 As Reference
Set reference11 = part1.CreateReferenceFromObject(line2D6)
'
Dim reference12 As Reference
Set reference12 = part1.CreateReferenceFromObject(line2D7)
'
Dim constraint5 As Constraint
Set constraint5 = constraints1.AddBiEltCst(catCstTypeAngle, reference11, reference12)
'
constraint5.Mode = catCstModeDrivingDimension
'
constraint5.AngleSector = catCstAngleSector2
'
Dim angle1 As Angle
Set angle1 = constraint5.Dimension
'
angle1.Value = 45#
'
Dim reference13 As Reference
Set reference13 = part1.CreateReferenceFromObject(line2D7)
'
Dim constraint6 As Constraint
Set constraint6 = constraints1.AddMonoEltCst(catCstTypeLength, reference13)
'
constraint6.Mode = catCstModeDrivingDimension
'
Dim length1 As Length
Set length1 = constraint6.Dimension
'
length1.Value = 10#
'
Dim line2D8 As Line2D
Set line2D8 = factory2D1.CreateLine(xinicial + passo - 7.071068, largura / 1.5 + 10 * espessura, xinicial + passo, largura / 1.5 + 10 * espessura - 7.071068)    '
'
line2D8.ReportName = 16
'
line2D8.StartPoint = point2D2
'
line2D8.EndPoint = point2D3
'
Dim reference14 As Reference
Set reference14 = part1.CreateReferenceFromObject(line2D3)
'
Dim reference15 As Reference
Set reference15 = part1.CreateReferenceFromObject(line2D8)
'
Dim constraint7 As Constraint
Set constraint7 = constraints1.AddBiEltCst(catCstTypeAngle, reference14, reference15)
'
constraint7.Mode = catCstModeDrivingDimension
```

```
constraint7.AngleSector = catCstAngleSector3
'
Dim angle2 As Angle
Set angle2 = constraint7.Dimension
'
angle2.Value = 45#
'
Dim reference16 As Reference
Set reference16 = part1.CreateReferenceFromObject(line2D8)
'
Dim constraint8 As Constraint
Set constraint8 = constraints1.AddMonoEltCst(catCstTypeLength, reference16)
'
constraint8.Mode = catCstModeDrivingDimension
'
Dim length2 As Length
Set length2 = constraint8.Dimension
'
length2.Value = 10#
'
Dim line2D9 As Line2D
Set line2D9 = factory2D1.CreateLine(xinicial, -largura / 1.5 - 10 * espessura + 7.071068, xinicial + 7.071068, -largura / 1.5 - 10 * espessura)    'linha 7
'
line2D9.ReportName = 17
'
line2D9.StartPoint = point2D7
'
line2D9.EndPoint = point2D6
'
Dim reference17 As Reference
Set reference17 = part1.CreateReferenceFromObject(line2D6)
'
Dim reference18 As Reference
Set reference18 = part1.CreateReferenceFromObject(line2D9)
'
Dim constraint9 As Constraint
'
Set constraint9 = constraints1.AddBiEltCst(catCstTypeAngle, reference17, reference18)
'
constraint9.Mode = catCstModeDrivingDimension
'
constraint9.AngleSector = catCstAngleSector0
'
Dim angle3 As Angle
Set angle3 = constraint9.Dimension
'
angle3.Value = 45#
'
Dim reference19 As Reference
Set reference19 = part1.CreateReferenceFromObject(line2D9)
'
Dim constraint10 As Constraint
Set constraint10 = constraints1.AddMonoEltCst(catCstTypeLength, reference19)
'
constraint10.Mode = catCstModeDrivingDimension
'
Dim length3 As Length
Set length3 = constraint10.Dimension
'
length3.Value = 10#
'
Dim line2D10 As Line2D
Set line2D10 = factory2D1.CreateLine(xinicial + passo - 7.071068, -largura / 1.5 - 10 * espessura, xinicial + passo, -largura / 1.5 - 10 * espessura + 7.071068)
'
line2D10.ReportName = 18
'
line2D10.StartPoint = point2D5
'
line2D10.EndPoint = point2D4
'
Dim reference20 As Reference
Set reference20 = part1.CreateReferenceFromObject(line2D5)
'
Dim reference21 As Reference
Set reference21 = part1.CreateReferenceFromObject(line2D10)
'
Dim constraint11 As Constraint
Set constraint11 = constraints1.AddBiEltCst(catCstTypeAngle, reference20, reference21)
'
constraint11.Mode = catCstModeDrivingDimension
'
constraint11.AngleSector = catCstAngleSector1
'
Dim angle4 As Angle
Set angle4 = constraint11.Dimension
'
angle4.Value = 45#
'
Dim reference22 As Reference
Set reference22 = part1.CreateReferenceFromObject(line2D10)
'
Dim constraint12 As Constraint
Set constraint12 = constraints1.AddMonoEltCst(catCstTypeLength, reference22)
'
constraint12.Mode = catCstModeDrivingDimension
'
Dim length4 As Length
Set length4 = constraint12.Dimension
'
length4.Value = 10#
'
'-----------------------------------> Acabando de dar forma ao exterior da matriz, procede-se à localização dos centros dos furos para aperto
'
'
Dim point2D9 As Point2D
Set point2D9 = factory2D1.CreatePoint(xinicial + 12 + 5, largura / 1.5 - 0.05 * largura) 'centro circunferencia 1 com descentramento
'
point2D9.ReportName = 19
'
Dim circle2D1 As Circle2D
Set circle2D1 = factory2D1.CreateClosedCircle(xinicial + 12 + 5, largura / 1.5 - 0.05 * largura, 4) 'circunferencia 1 com descentramento
```

```
'
circle2D1.CenterPoint = point2D9
'
circle2D1.ReportName = 20
'
Dim point2D10 As Point2D
Set point2D10 = factory2D1.CreatePoint(xinicial + passo - 12 - 5, largura / 1.5)  'centro circunferencia 2
'
point2D10.ReportName = 21
'
Dim circle2D2 As Circle2D
Set circle2D2 = factory2D1.CreateClosedCircle(xinicial + passo - 12 - 5, largura / 1.5, 4)   'circunferencia 2
'
circle2D2.CenterPoint = point2D10
'
circle2D2.ReportName = 22
'
Dim point2D11 As Point2D
Set point2D11 = factory2D1.CreatePoint(xinicial + 12 + 5, -largura / 1.5)    'centro circunferencia 3
'
point2D11.ReportName = 23
'
Dim circle2D3 As Circle2D
Set circle2D3 = factory2D1.CreateClosedCircle(xinicial + 12 + 5, -largura / 1.5, 4)    'circunferencia 3
'
circle2D3.CenterPoint = point2D11
'
circle2D3.ReportName = 24
'
Dim point2D12 As Point2D
Set point2D12 = factory2D1.CreatePoint(xinicial + passo - 12 - 5, -largura / 1.5)   'centro circunferencia 4
'
point2D12.ReportName = 25
'
Dim circle2D4 As Circle2D
Set circle2D4 = factory2D1.CreateClosedCircle(xinicial + passo - 12 - 5, -largura / 1.5, 4)  'circunferencia 4
'
circle2D4.CenterPoint = point2D12
'
circle2D4.ReportName = 26
'
'----------> No caso de a matriz ser muito comprida, é feito mais um furo ao centro, graças ao comando:
'
If passo > 100 Then
    '
    Dim point2D13 As Point2D
    Set point2D13 = factory2D1.CreatePoint(xinicial + passo / 2, largura / 1.5) 'centro circunferencia 5
    '
    point2D13.ReportName = 27
    '
    Dim circle2D5 As Circle2D
    Set circle2D5 = factory2D1.CreateClosedCircle(xinicial + passo / 2, largura / 1.5, 4) 'circunferencia 5
    '
    circle2D5.CenterPoint = point2D13
    '
    circle2D5.ReportName = 28
    '
    ''''''''''''''''''''''''''''''''''''''''''
    '
    Dim point2D14 As Point2D
    Set point2D14 = factory2D1.CreatePoint(xinicial + passo / 2, -largura / 1.5) 'centro circunferencia 6
    '
    point2D14.ReportName = 29
    '
    Dim circle2D6 As Circle2D
    Set circle2D6 = factory2D1.CreateClosedCircle(xinicial + passo / 2, -largura / 1.5, 4) 'circunferencia 6
    '
    circle2D6.CenterPoint = point2D14
    '
    circle2D6.ReportName = 30
End If
'
'
sketch1.CloseEdition                '----------> Encerramento da edição do Sketch
'
part1.InWorkObject = body1
'
part1.Update
'
part1.InWorkObject = body1
'
Dim shapeFactory1 As ShapeFactory
Set shapeFactory1 = part1.ShapeFactory
'
Dim pad1 As Pad
Set pad1 = shapeFactory1.AddNewPad(sketch1, 12#)    '----------> Extrusão da espessura da matriz
'
pad1.DirectionOrientation = catInverseOrientation
'
Dim limit1 As Limit
Set limit1 = pad1.FirstLimit
'
Dim length5 As Length
Set length5 = limit1.Dimension
'
'----------> O valor da espessura da matriz tem duas possibilidades mediante a espessura da banda seja superior ou inferior a 1 milimetro
'
If espessura < 1 Then
    length5.Value = 25
Else
    length5.Value = 35
End If
'
part1.Update
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'POCKET para as cabeças dos parafusos de aperto
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
Set documents1 = CATIA.Documents
'
Set partDocument1 = documents1.Item("Part" & NUM & ".CATPart")
```

```
'
Set part1 = partDocument1.Part
'
Set bodies1 = part1.Bodies
'
Set body1 = bodies1.Item("PartBody")
'
part1.InWorkObject = body1
'
Set sketches1 = body1.Sketches
'
Set hybridBodies1 = part1.HybridBodies
'
Set hybridBody1 = hybridBodies1.Item("Geometrical Set.1")
'
Set hybridShapes1 = hybridBody1.HybridShapes
'
Set reference1 = hybridShapes1.Item("Plane.1")
'
Set sketch1 = sketches1.Add(reference1)
'
arrayOfVariantOfDouble1(0) = 0#
arrayOfVariantOfDouble1(1) = 0#
arrayOfVariantOfDouble1(2) = -espessura - 0.75
arrayOfVariantOfDouble1(3) = 1#
arrayOfVariantOfDouble1(4) = 0#
arrayOfVariantOfDouble1(5) = 0#
arrayOfVariantOfDouble1(6) = 0#
arrayOfVariantOfDouble1(7) = 1#
arrayOfVariantOfDouble1(8) = 0#
Set sketch1Variant = sketch1
'
part1.InWorkObject = sketch1
'
Set factory2D1 = sketch1.OpenEdition()
'
Set geometricElements1 = sketch1.GeometricElements

'
Set axis2D1 = geometricElements1.Item("AbsoluteAxis")
'
Set line2D1 = axis2D1.GetItem("HDirection")
'
line2D1.ReportName = 1
'
Set line2D2 = axis2D1.GetItem("VDirection")
'
line2D2.ReportName = 2
'
Set point2D1 = factory2D1.CreatePoint(xinicial + 12 + 5, largura / 1.5 - 0.05 * largura)    'centro circunferencia 1
'
point2D1.ReportName = 3
'
Set circle2D1 = factory2D1.CreateClosedCircle(xinicial + 12 + 5, largura / 1.5 - 0.05 * largura, 7) 'circunferencia 1
'
circle2D1.CenterPoint = point2D1
'
circle2D1.ReportName = 4
'
Set point2D2 = factory2D1.CreatePoint(xinicial + passo - 12 - 5, largura / 1.5) 'centro circunferencia 2
'
point2D2.ReportName = 5
'
Set circle2D2 = factory2D1.CreateClosedCircle(xinicial + passo - 12 - 5, largura / 1.5, 7)  'circunferencia 2
'
circle2D2.CenterPoint = point2D2
'
circle2D2.ReportName = 6
'
Set point2D3 = factory2D1.CreatePoint(xinicial + 12 + 5, -largura / 1.5)    'centro circunferencia 3
'
point2D3.ReportName = 7
'
Set circle2D3 = factory2D1.CreateClosedCircle(xinicial + 12 + 5, -largura / 1.5, 7) 'circunferencia 3
'

circle2D3.CenterPoint = point2D3
'
circle2D3.ReportName = 8
'
Set point2D4 = factory2D1.CreatePoint(xinicial + passo - 12 - 5, -largura / 1.5)    'centro circunferencia 4
'
point2D4.ReportName = 9
'
Set circle2D4 = factory2D1.CreateClosedCircle(xinicial + passo - 12 - 5, -largura / 1.5, 7) 'circunferencia 4
'
circle2D4.CenterPoint = point2D4
'
circle2D4.ReportName = 10
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
If passo > 100 Then
    Set point2D5 = factory2D1.CreatePoint(xinicial + passo / 2, largura / 1.5)  'centro circunferencia 5
    '
    point2D5.ReportName = 11
    '
    Set circle2D5 = factory2D1.CreateClosedCircle(xinicial + passo / 2, largura / 1.5, 7) 'circunferencia 5
    '
    circle2D5.CenterPoint = point2D5
    '
    circle2D5.ReportName = 12
    ''''''''''''''''''''''''''''''''''''''''''
    Set point2D6 = factory2D1.CreatePoint(xinicial + passo / 2, -largura / 1.5)  'centro circunferencia 6
    '
    point2D6.ReportName = 13
    '
    Set circle2D6 = factory2D1.CreateClosedCircle(xinicial + passo / 2, -largura / 1.5, 7) 'circunferencia 6
    '
    circle2D6.CenterPoint = point2D6
    '
    circle2D6.ReportName = 14
End If
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
sketch1.CloseEdition
```

```
'
part1.InWorkObject = body1
'
part1.Update
'
Set shapeFactory1 = part1.ShapeFactory
'
Dim pocket1 As Pocket
Set pocket1 = shapeFactory1.AddNewPocket(sketch1, 8#)
'
Set limit1 = pocket1.FirstLimit
'
Set length1 = limit1.Dimension
'
length1.Value = 8#  'valor do POCKET
'
part1.Update
'
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'FUROS PARA SAÍDA DO RETRAÇO NA MATRIZ
'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
Dim CONTA As Integer
'
CONTA = InputBox("Quantos cortes existem nesta matriz?")
'
For i = 1 To CONTA                                              '-----> Ínicio de um ciclo FOR
    Set documents1 = CATIA.Documents
    '
    Set partDocument1 = documents1.Item("Part" & NUM & ".CATPart")
    '
    Set part1 = partDocument1.Part
    '
    Set bodies1 = part1.Bodies
    '
    Set body1 = bodies1.Item("PartBody")
    '
    part1.InWorkObject = body1
    '
    Set productDocument1 = CATIA.ActiveDocument
    '
    Dim selection1 As Selection
    Set selection1 = productDocument1.Selection
    '
    selection1.Clear
    '
    Dim partDocument2 As PartDocument
    Set partDocument2 = documents1.Item("Strip_Layout.CATPart")
    '
    Dim part2 As Part
    Set part2 = partDocument2.Part
    '
    Dim bodies2 As Bodies
    Set bodies2 = part2.Bodies
    '
    Dim body2 As Body
    Set body2 = bodies2.Item("Strip Layout")
    '
    Set sketches1 = body2.Sketches
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    Dim NOME As String                                          '''----------> Declaração de uma variável STRING que conterá o nome de um sketch
    NOME = InputBox("Qual o SKETCH que contém o corte?")        '''
    '''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
    '
    Set sketch1 = sketches1.Item(NOME & ".S")
    '
    selection1.Add sketch1
    '
    selection1.Copy
    '
    Set productDocument1 = CATIA.ActiveDocument
    '
    Dim selection2 As Selection
    Set selection2 = productDocument1.Selection
    '
    selection2.Clear
    '
    selection2.Add body1
    '
    selection2.Paste
    '
    Dim sketches2 As Sketches
    Set sketches2 = body1.Sketches
    '
    Dim sketch2 As Sketch
    Set sketch2 = sketches2.Item(NOME & ".S")
    '
    Set geometricElements1 = sketch2.GeometricElements
    '
    Set axis2D1 = geometricElements1.Item("AbsoluteAxis")
    '
    part1.Inactivate axis2D1
    '
    part1.Update
    '
    Set shapeFactory1 = part1.ShapeFactory
    '
    Set pocket1 = shapeFactory1.AddNewPocket(sketch2, 35 + espessura)
    '
    part1.Update
    '
    Set productDocument1 = CATIA.ActiveDocument
    '
    Dim selection3 As Selection
    Set selection3 = productDocument1.Selection
    '
    selection3.Clear
    '
    Dim sketch3 As Sketch
    Set sketch3 = sketches1.Item("Furo_" & NOME & ".S")
    '
```

```
    selection3.Copy
    '
    part1.InWorkObject = body1
    '
    Set productDocument1 = CATIA.ActiveDocument
    '
    Dim selection4 As Selection
    Set selection4 = productDocument1.Selection
    '
    selection4.Clear
    '
    selection4.Add body1
    '
    selection4.Paste
    '
    Dim sketch4 As Sketch
    Set sketch4 = sketches2.Item("Furo_" & NOME & ".S")
    '
    Dim geometricElements2 As GeometricElements
    Set geometricElements2 = sketch4.GeometricElements
    '
    Dim axis2D2 As Axis2D
    Set axis2D2 = geometricElements2.Item("AbsoluteAxis")
    '
    part1.Inactivate axis2D2
    '
    part1.Update
    '
    Dim pocket2 As Pocket
    Set pocket2 = shapeFactory1.AddNewPocket(sketch4, 35 + espessura)
    '
    part1.Update
    '
Next                                                                    '----->Fim do ciclo FOR
'
End Sub
'                                                                   '----------> Fim do programa
```